CERTIFIED COPY
OF
PRIORITY APPLICATION

Ministero delle Atti

Direzione Generale per lo Svilupp

Ufficio Italiano Brevetti e Marchi

Ufficio G2

---

utenticazione di copia di documenti relativi al brevetto per: INVENZIONE INDUSTRIALE
N. 1302431 rilasciato il 05/09/2000 (RM 1998 A 000542 del 12.08.1998)

Si dichiara che l'unita copia è conforme ai documenti originali conservati dall'ufficio.

Ad esclusione del prospetto A (pag. 2).

oma, li..... 28 SET. 2004

IL FUNZIONARIO

..... Giampietro Carlotto .....

Giampietro Carlotto

AL MINISTERO DELL'INDUSTRIA DEL COMMERCIO E DELL'ARTIGIANATO **MODULO A**

UFFICIO ITALIANO BREVETTI E MARCHI - ROMA

DOMANDA DI BREVETTO PER INVENZIONE INDUSTRIALE, DEPOSITO RISERVE, ANTICIPATA ACCESSIBILITÀ AL PUBBLICO

A. RICHIEDENTE (I)

1) Denominazione: ALASI di Arcieri Franco & C. s.a.s. SA

Residenza: Austis (Nuoro), ITALIA codice: 00915950919

2) Denominazione:

Residenza: codice:

B. RAPPRESENTANTE DEL RICHIEDENTE PRESSO L'U.I.B.M.

cognome e nome: de Benedetti Fabrizio ed altri cod. fiscale:

denominazione studio di appartenenza: SOCIETA' ITALIANA BREVETTI S.p.A.

via: Piazza di Pietra n.: 0039 città: ROMA cap: 00186 (prov): RM

C. DOMICILIO ELETTIVO destinatario

via: n.: città: cap: (prov):

D. TITOLO classe proposta (sez/cl/scl): gruppo/sottogruppo:

"DISPOSITIVO DI CONTROLLO DI ACCESSI IN RETE TRAMITE IL RICONOSCIMEN= TO VELOCE DI TRAME APPLICATIVE CHE SODDISFANO UN INSIEME DI REGOLE PREDEFINITE"

ANTICIPATA ACCESSIBILITÀ AL PUBBLICO: SI [ ] NO [X] **SE ISTANZA: DATA** / / **N° PROTOCOLLO**

E. INVENTORI DESIGNATI

cognome nome		cognome nome	
1) ARCIERI Franco		3) TALAMO Maurizio	
2) MARINELLI Guido Maria		4)	

F. PRIORITÀ

	nazione o organizzazione	tipo di priorità	numero di domanda	data di deposito	allegato S/R
1)				/ /	
2)				/ /	

SCIOGLIMENTO RISERVE

Data	N° Protocollo

G. CENTRO ABILITATO DI RACCOLTA COLTURE DI MICRORGANISMI, denominazione

H. ANNOTAZIONI SPECIALI

DOCUMENTAZIONE ALLEGATA

	N. es.			
Doc. 1)	2	PROV	n. pag. 85	riassunto con disegno principale, descrizione e rivendicazioni (obbligatorio 1 esemplare)
Doc. 2)	2	PROV	n. tav. 12	disegno (obbligatorio se citato in descrizione, 1 esemplare
Doc. 3)	0	RIS		lettera d'incarico, procura o riferimento a procura generale
Doc. 4)	0	RIS		designazione inventore
Doc. 5)	0	RIS		documenti di priorità con traduzione in italiano
Doc. 6)	0	RIS		autorizzazione o atto di cessione
Doc. 7)	0			nominativo completo del richiedente

SCIOGLIMENTO RISERVE

Data	N° Protocollo

confronta singole priorità

8) attestati di versamento, totale lire novecentoquindicimila= obbligatorio

COMPILATO IL 12 08 1998 **FIRMA DEL (I) RICHIEDENTE (I)**

Giorgio Strini

(Iscr. Albo n. 452 BM)

CONTINUA SI/NO NO

DEL PRESENTE ATTO SI RICHIEDE COPIA AUTENTICA SI/NO SI

UFFICIO PROVINCIALE IND. COMM. ART. DI ROMA codice 58

VERBALE DI DEPOSITO NUMERO DI DOMANDA R M 98 A 000542 Reg. A

L'anno millenovecento novantotto, il giorno dodici, del mese di agosto

il (i) richiedente (i) sopraindicato (i) ha (hanno) presentato a me sottoscritto la presente domanda, corredate di n. 00 fogli aggiuntivi per la concessione del brevetto soprariportato.

I. ANNOTAZIONI VARIE DELL'UFFICIO ROGANTE

timbro dell'ufficio

IL DEPOSITANTE

L'UFFICIALE ROGANTE

CESARE A. GASPERINI

Funzionario Camerale

RM 98 A 000542

SIB 91758

DESCRIZIONE dell'invenzione industriale dal titolo:
"DISPOSITIVO DI CONTROLLO DI ACCESSI IN RETE TRAMITE IL RICONOSCIMENTO VELOCE DI TRAME APPLICATIVE CHE SODDISFANO UN INSIEME DI REGOLE PREDEFINITE"
della ditta italiana ALASI di Arcieri Franco & C. s.a.s.
con sede in AUSTIS (NUORO) - ITALIA

-!-!-!-

DESCRIZIONE

La presente invenzione ha come oggetto un dispositivo di controllo di accessi in rete tramite il riconoscimento veloce di trame applicative che soddisfano un insieme di regole predefinite.

In particolare, il dispositivo secondo l'invenzione permette sia il rilevamento e l'interpretazione di protocolli applicativi di sistemi di trasmissione dati su rete sia il confronto di ogni trama di comunicazione rilevata ed interpretata con un insieme di "pattern" di controllo. Nel caso di trama riconosciuta, il dispositivo permette l'accesso al servizio. Nel caso di trama non riconosciuta, il dispositivo nega l'accesso al servizio.

Per "pattern" (o regola d'accesso) verrà intesa

all'interno della presente descrizione la specifica di riconoscimento di una particolare trama di comunicazione.

Tale specifica verrà preferenzialmente intesa come un insieme di coppie <tipo di dato>/<valore dato> assunte dai campi all'interno della trama di comunicazione. Le coppie <tipo di dato>/<valore dato> sono specificate in base ai vari livelli di comunicazione che si trovano all'interno della trama di comunicazione sia per quanto concerne la parte di controllo sia anche per quanto concerne la parte di informazione. Nel corso della presente descrizione verranno illustrate, a titolo di esempio, trame di comunicazione di tipo HTTP (i servizi di browsing su Internet).

Dispositivi di controllo di accessi in rete sono noti e si possono suddividere in due grandi categorie:
1) In una prima categoria, le varie regole di accesso vengono rappresentate tramite matrici multidimensionali rappresentate in forma non-compressa, facendo uso di semplici linguaggi di accesso a tali matrici. Lo svantaggio di una tale rappresentazione è dato dalla notevole occupazione in memoria: una matrice 10-dimensionale con 100 elementi per dimensione ha una occupazione di memoria di 100^{10}.

2) In una seconda categoria, le varie regole di accesso vengono rappresentate tramite matrici multidimensionali rappresentate in forma compressa. L'accesso a tali matrici non è di tipo diretto. Una tale modalità presenta lo svantaggio di richiedere l'utilizzo di linguaggi ad alto livello i quali, mediante operatori di test e di confronto, determinano la particolare procedura da attivare in risposta al riconoscimento di una regola d'accesso. Le particolari strutture di controllo così utilizzate appesantiscono il processo di interpretazione, rendendolo inefficiente. Risulta peraltro difficile, se non impossibile, la realizzazione di metodi generalizzati per il riconoscimento delle strutture informative su tecnologie veloci (firmware).

In ambedue le tipologie di dispositivi di tecnica nota sussiste poi l'ulteriore svantaggio che il riconoscimento delle trame di comunicazione non può essere basato su un qualsiasi componente della trama, ma solamente su trame a livello non applicativo.

La presente invenzione ovvia a tali problemi di tecnica precedente, in quanto prevede un dispositivo di controllo di accessi in rete tramite il riconoscimento deterministico di trame applicative che soddisfano un insieme di regole predefinite

comprendente:

- mezzi di rilevamento ed interpretazione delle trame applicative da riconoscere;
- mezzi di memorizzazione di regole predefinite;
- mezzi di compilazione delle regole predefinite in una struttura dati ad accesso diretto;
- mezzi di memorizzazione di detta struttura dati ad accesso diretto; e
- mezzi di confronto tra le trame applicative da riconoscere e detta struttura dati ad accesso diretto,

in cui il riconoscimento può essere effettuato su qualsiasi componente della trama ed in cui la struttura dati ad accesso diretto permette di ottenere un tempo di accesso sostanzialmente indipendente dalla numerosità delle regole.

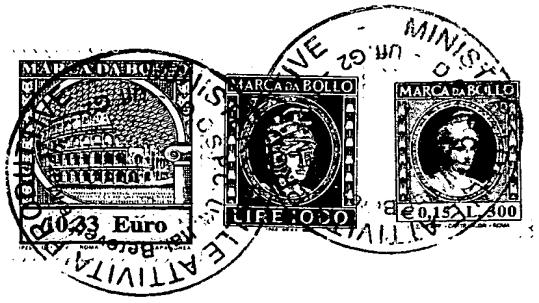
Come noto, una struttura dati ad accesso diretto ha la proprietà di consentire l'accesso all'elemento i-esimo senza dover necessariamente accedere preventivamente agli elementi precedenti, come avviene invece nel caso di strutture dati ad accesso sequenziale. Esempi noti di strutture dati ad accesso diretto sono i vettori, le matrici, le tavole di corrispondenza, la memoria di un elaboratore etc.

Un primo vantaggio del dispositivo di controllo di accessi secondo la presente invenzione è dato dalla

flessibilità con la quale è possibile realizzare un pattern di riconoscimento. Grazie all'utilizzo dell'apparecchio per il rilevamento e l'interpretazione di protocolli applicativi, spiegato in dettaglio nel seguito, il riconoscimento delle trame di comunicazione può essere infatti basato su una qualsiasi componente della trama, sia sulla parte controllo che sulla parte informazione. E' quindi possibile realizzare pattern di riconoscimento (e quindi limitare l'accesso) in base ai contenuti informativi scambiati e non soltanto in base agli indirizzi ed ai servizi di rete utilizzati.

Un secondo vantaggio del dispositivo secondo la presente invenzione è dato dalla sua capacità di gestire un elevatissimo numero di pattern (dell'ordine dei milioni) senza alcun degrado delle prestazioni.

In un contesto nel quale siano coinvolti un numero elevato di utenti, di server e di servizi applicativi sui server stessi, quando si vuole gestire in modo diretto l'accessibilità di ogni utente al singolo server e servizio applicativo da questo reso disponibile ci si trova infatti in presenza di una crescita quadratica dei pattern. Ad esempio, dati 1000 utenti sul territorio e 100 server di cui si vuole gestire e controllare gli accessi, vengono generati



1000*100=100.000 pattern. Questo numero aumenta ancora nel caso in cui si voglia gestire e controllare l'accesso alle applicazioni per ogni singolo server, rendendo, in casi reali su organizzazioni medio/grandi, il numero di pattern stimabile nell'ordine dei milioni.

Tale numero di pattern è del tutto accettabile per il dispositivo di cui alla presente invenzione.

Il riconoscimento delle trame di comunicazione acquisite è infatti basato su un algoritmo deterministico (quindi né euristico né probabilistico) di accesso in grado di garantire un tempo di accesso costante ed indipendente (sotto qualsiasi input) dal numero dei pattern.

Per ogni trama correttamente riconosciuta, il dispositivo di controllo di accessi esegue poi l'operazione di coordinamento associata al riconoscimento. A riconoscimento avvenuto infatti, il dispositivo attiverà una comunicazione a livello TCP/IP (o livelli corrispondenti di altri protocolli) con l'applicazione server individuata come risultato del riconoscimento, anche parziale, della componente informativa della trama di ingresso, fornendo come parametri parte della componente informativa già riconosciuta oppure non ancora processata. Le modalità

di invio (formato di invio dei parametri, numero dei parametri da inviare, applicazione da attivare etc.) sono associate all'azione di riconoscimento e sono quindi memorizzate nei pattern.

Il dispositivo di controllo di accessi secondo la presente invenzione può essere configurato per operare tanto in logica diretta che in logica negata.

In logica diretta saranno considerate accettate, e quindi portate a destinazione, tutte le trame che soddisfano i pattern di riconoscimento.

In logica negata saranno considerate accettate, e quindi portate a destinazione, tutte le trame che non soddisfano i pattern di riconoscimento. Tutte le trame riconosciute non saranno portate a destinazione.

La presente invenzione verrà qui di seguito descritta tramite una sua forma di realizzazione preferita, illustrata a scopo esemplificativo e non limitativo. Verrà fatto riferimento alle figure dei disegni allegati, in cui:

la figura 1 mostra un diagramma schematico dello standard OSI;

la figura 2 mostra una rappresentazione schematica del tipo di dati utilizzati nelle comunicazioni su rete;

la figura 3 mostra una rappresentazione

schematica del tipo di dati utilizzati nelle comunicazioni su rete con riferimento al protocollo TCP/IP;

la figura 4 mostra uno schema a blocchi dell'apparecchio di rilevamento ed interpretazione facente parte del dispositivo di controllo di accessi secondo la presente invenzione;

la figura 5 mostra uno schema di flusso che spiega il funzionamento della componente di figura 4;

le figure 6 e 7 mostrano ulteriori schemi di flusso per la comprensione di quanto descritto con riferimento alla figura 5;

le figure 8A e 8B mostrano un esempio di albero applicativo arricchito di informazioni di tipo statistico ottenuto tramite la componente di figura 4;

la figura 9 mostra uno schema a blocchi del dispositivo di controllo di accessi secondo la presente invenzione;

le figure 10A e 10B mostrano schemi esemplificativi della corrispondenza logica tra grafo bipartito e matrice bidimensionale;

la figura 11 contiene un esempio di specifica di regole predefinite; e

la figura 12 mostra una rappresentazione in forma matriciale di sequenze di identificatori numerici.

La trasmissione di dati da un dispositivo di sorgente ad un dispositivo di destinazione può avvenire secondo modalità differenti. Al fine di assicurare uno scambio di dati con una ampia probabilità di mancanza di errori è però necessario adottare un insieme di regole o procedure di controllo. Tali regole o procedure sono note con il termine di "protocolli di comunicazione".

Un protocollo standard di comunicazione è l'"Open System Interconnection" (OSI) della International Standards Organization (ISO). Tale protocollo è organizzato secondo una suddivisione in sette livelli, mostrata in figura 1. Il livello 7 (applicazione) del lato sorgente contiene informazioni relative al semplice messaggio (M) da inviare verso il lato destinazione. I successivi livelli del lato sorgente aggiungono informazioni di controllo al messaggio: il livello 6 (presentazione) suddivide i dati del messaggio originale in blocchi (M1 ed M2); il livello 5 (sessione) aggiunge un titolo (S) per indicare il mittente, il destinatario ed alcune informazioni relative alla sequenza; il livello 4 (trasporto) aggiunge informazioni (T) relative alla connessione logica tra il mittente ed il destinatario; il livello 3 (rete) aggiunge informazioni relative al percorso

(N) ed il messaggio viene suddiviso in pacchetti che rappresentano l'unità standard di comunicazione in una rete; il livello 2 (collegamento dati) aggiunge una parte di titolo (B) ed una parte di coda (E) al messaggio per assicurare il corretto ordine dei vari pacchetti e correggere errori di trasmissione; i singoli bit del messaggio e delle informazioni di controllo via via aggiunte dai vari livelli vengono trasmessi sul mezzo fisico attraverso il livello 1. La freccia F1 verso il basso sul lato sorgente indica le modalità secondo le quali viene costruito il messaggio in partenza. Tutte le aggiunte al messaggio vengono verificate e rimosse dal corrispondente livello dal lato del destinatario. La freccia F2 verso l'alto sul dato destinatario indica le modalità secondo le quali viene ricostruito il messaggio in arrivo.

Facendo riferimento allo standard OSI, l'unità di comunicazione in una rete è il pacchetto. I pacchetti solo a loro volta suddivisi in frame. L'inizio e la fine di ciascun frame vengono in genere stabiliti tramite caratteri di delimitazione. I frame sono a loro volta suddivisi in frame di informazione e frame di controllo. I frame di informazione servono al trasporto di dati relativi al messaggio da trasmettere lungo la rete, mentre i frame di controllo servono a

gestire le modalità secondo le quali tale trasporto deve avvenire, vale a dire al controllo del flusso ed all'attivazione delle azioni di recupero degli errori. Sia i frame di informazione che i frame di controllo contengono una parte di intestazione che identifica il tipo di frame ed una parte di corpo tipica invece del frame stesso.

La struttura dei frame di informazione verrà descritta facendo riferimento alla figura 2. Nella parte superiore di tale figura è rappresentata in maniera schematica la struttura generica di un pacchetto di livello OSI 2, comprendente cioè sia frame di informazione 1 che frame di controllo 2. La costituzione di un singolo frame di informazione (livello OSI 3) indica la presenza di una parte di intestazione 3, che contiene l'identificazione che il frame in oggetto è un frame di informazione, e di una parte di corpo 4. La parte di corpo (livelli OSI 4-7) contiene il messaggio 5 vero e proprio, unitamente ad una serie 6 di campi, rappresentati in maniera esemplificativa in figura con i caratteri C1, C2 e C3, tipici della particolare sintassi applicativa utilizzata. Per sintassi applicativa si intendono le informazioni relative al numero di campi contenuti all'interno della serie 6, al significato di ciascuno

di tali campi ed ai dati in essi contenuti.

Il modello OSI fin qui schematicamente riassunto è solamente un modello concettuale. Uno dei protocolli attualmente più utilizzati ed universalmente accettati è il protocollo TCP/IP (Transmission Control Protocol and Internet Protocol). Tale protocollo, come anche altri protocolli di comunicazione adottati, è spiegabile tramite riferimenti alla struttura a livelli del modello OSI. In ciascuno di tali protocolli infatti, un determinato livello di sorgente suddividerà i dati che riceve da un livello superiore aggiungendo agli stessi una intestazione e/o una coda per poi passare il tutto ad un livello inferiore. Dal lato destinazione avverranno le operazioni inverse.

Con riferimento alla successiva figura 3, viene mostrata una rappresentazione schematica del tipo di dati utilizzati nelle comunicazioni su rete locale con riferimento al protocollo TCP/IP trasportante il servizio applicativo HTTP (browsing su Internet).

Il livello Ethernet comprende sostanzialmente quattro tipi di campi:

- un campo 101 di indirizzo della scheda di rete di destinazione;
- un campo 102 di indirizzo della scheda di rete di sorgente;

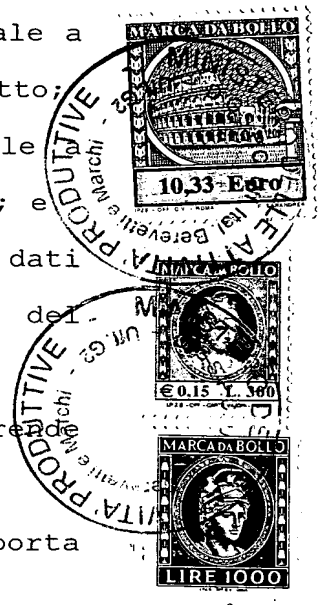
- un campo 103 indicativo del protocollo di comunicazione trasportato, in questo caso indicativo del protocollo IP e della lunghezza della parte informazione; e
- un campo informazione 104, contenente cioè i dati del livello Ethernet, vale a dire tutta la struttura del protocollo IP trasportato.

Il livello IP (incapsulato nel livello Ethernet) comprende sostanzialmente sei tipi di campi:

- una serie di campi di controllo 105 identificativi della versione, della lunghezza, delle opzioni di trasmissione, filler etc.;
- un campo 106 indicativo del protocollo di comunicazione trasportato, in questo caso indicativo del protocollo TCP;
- un campo 107 di indirizzo IP di destinazione, vale a dire l'indirizzo IP di chi deve ricevere il pacchetto;
- un campo 108 di indirizzo IP di sorgente, vale a dire l'indirizzo IP di chi ha inviato il pacchetto; e
- un campo informazione 109, contenente cioè i dati del livello IP, vale a dire tutta la struttura del protocollo TCP trasportato.

Il livello TCP (incapsulato nel livello IP) comprende quattro tipi di campi:

- un campo porta di sorgente 110, indicante la porta



di servizio TCP utilizzata da chi ha inviato il pacchetto;

- un campo porta di destinazione 111, indicante la porta di servizio TCP utilizzata dal ricevitore del pacchetto;

- una serie di campi di controllo 112 identificanti l'ID di pacchetto, la finestra di lavoro, il crc, opzioni varie etc.; e

- un campo di informazione 113, contenente cioè i dati del livello TCP, vale a dire tutta la struttura del servizio applicativo HTTP trasportato, cioè i comandi del linguaggio HTTP e, nella sua parte informativa, i comandi del linguaggio HTML.

Sistemi di rilevamento dei dati trasmessi tra un nodo sorgente ed un nodo di destinazione sono già noti. Tali sistemi si limitano però all'analisi dei livelli OSI 2 (collegamento dati) e OSI 3 (rete). Il rilevamento e la successiva interpretazione dei dati a tali livelli permettono soltanto l'individuazione di anomalie nel protocollo di scambio tra i vari componenti di un sistema di trasmissione dati su rete.

Uno svantaggio tipico di tali sistemi di tecnica precedente è pertanto l'impossibilità di decodificare l'informazione di tipo applicativo trasportata sulla rete, vale a dire l'informazione relativa ai livelli

4-7 dello standard OSI.

Nelle successive figure da 4 ad 8B verranno descritte nel dettaglio la struttura ed il funzionamento di un apparecchio per il rilevamento e l'interpretazione di protocolli applicativi.

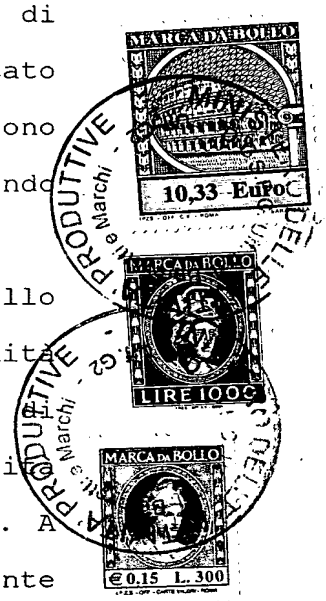
Verrà fatto ora riferimento alla figura 4, che mostra uno schema a blocchi dell'apparecchio. In tale figura vengono innanzitutto mostrati un nodo di sorgente 7 ed un nodo di destinazione 8, terminali del tratto di rete i cui dati vengono rilevati ed interpretati. Lungo il collegamento tra tali due nodi, rappresentato schematicamente dalle frecce F3, F4, F5, F6 e dal mezzo trasmissivo 23, viaggiano in maniera bidirezionale dati relativi a più comunicazioni tra un primo insieme di elaboratori di sorgente (non indicati in figura) a monte del nodo di sorgente 7 ed un secondo insieme di elaboratori di destinazione (non indicati in figura) a valle del nodo di destinazione 8.

Tali dati vengono rilevati tramite un dispositivo 9 di rilevamento dati. Diversi sono i dispositivi di rilevamento del tipo noti sul mercato; per quanto riguarda le reti basate su tecnologia Ethernet è ad esempio possibile citare la scheda Fast Etherlink XL™ della ditta 3Com™. Per quanto riguarda le reti basate

su tecnologia X.25 è possibile citare ad esempio la scheda S508 della ditta canadese Sangoma™. Tale scheda può operare con diversi standard a livello OSI 1 (livello fisico) quali ad esempio lo standard RS232 (o V.24) e lo standard RS422 (o V.35). Gli standard di livello OSI 2 (collegamento dati) con i quali tale scheda può operare sono ad esempio lo standard HDLC oppure, contenuto in questo, lo standard X.25. Il tipo di dispositivo di rilevamento 9 da scegliersi ai fini della presente invenzione potrà comunque variare a seconda degli standard di livello OSI 1 o OSI 2 sui quali si desideri operare. Sarà infatti possibile pensare di utilizzare dispositivi di rilevamento che operino con standard implementativi differenti del livello OSI 2, quali ad esempio il "Frame Relay" o lo SDLC o ancora il BSC o altri similari. Tali dispositivi sono comunque ben noti all'esperto del ramo e non verranno qui discussi in dettaglio.

Il rilevamento avviene in maniera "trasparente" tramite due connettori paralleli 10 ed 11, indicati schematicamente in figura, atti a permettere il rilevamento dei dati provenienti rispettivamente dal nodo di sorgente 7 e dal nodo di destinazione 8. Il dispositivo di rilevamento 9, mostrato nel suo complesso nel blocco indicato a tratteggio in figura,

è tale da comprendere un ricevitore dei dati di sorgente 12, un ricevitore dei dati di destinazione 13 ed una interfaccia di connessione 14. Il ricevitore dei dati di sorgente 12 è tale da permettere la ricezione dei soli dati provenienti dal nodo di sorgente 7 secondo quanto schematicamente indicato dalla freccia F7; il ricevitore dei dati di destinazione 13 è tale invece da permettere la ricezione dei soli dati provenienti dal nodo di destinazione 8 secondo quanto schematicamente indicato dalla freccia F8. I dati così ricevuti vengono trasmessi alla interfaccia di connessione 14, secondo quanto indicato dalle frecce F9 ed F10.



Ciascun pacchetto di dati ad un livello corrispondente al livello OSI 2 letto tramite l'unità di rilevamento 9 viene inviato ad una unità di controllo 15, come indicato dalla freccia F11. L'unità di controllo 15 verrà descritta oltre in dettaglio. A ciascuno di tali pacchetti viene associato un istante temporale di lettura tramite una unità 16 di datazione, rappresentata per comodità di presentazione all'esterno dell'unità di controllo 15 e collegata a quest'ultima come indicato tramite la freccia F12. Tale unità 16 di datazione può essere un qualsiasi dispositivo a tempo assoluto presente in commercio, in

particolare via radio o satellitare. Nella modalità di realizzazione preferita delle presente invenzione si è utilizzato un orologio digitale radiocontrollato che si tara sull'ora CET (Central European Time) irradiata tramite satellite geostazionario.

Successivamente all'associazione dell'istante temporale di lettura tramite l'unità 16 di datazione, l'unità di controllo 15 provvede a riordinare logicamente i singoli frame in modo da ricostruire l'esatta sequenza logico/temporale di spedizione dei frame che, come noto, non sempre coincide con la sequenza di ricezione: è infatti possibile, causa le tecniche di instradamento su reti di telecomunicazione, che una sequenza di spedizione del tipo "ABC" possa essere ricevuta in tutte e sei le sue possibili combinazioni, vale a dire "ABC", "ACB", "BAC", "BCA", "CAB", "CBA". L'unità di controllo 15 provvede quindi alla discriminazione dei frame di informazione dai frame di controllo. Nel caso in cui l'informazione venga ad esempio trasmessa in HDLC, l'ultimo bit della parte di intestazione di un frame di informazione è 0 mentre l'ultimo bit della parte di intestazione di un frame di controllo è 1. All'interno dell'unità di controllo 15 sono pertanto presenti mezzi, non indicati in figura, atti alla

discriminazione di tale ultimo bit, ad esempio un firmware contenuto in una ROM. In ogni caso, qualunque sia il codice di trasmissione dati utilizzato, saranno sempre note le modalità che distinguono un frame di controllo da un frame di informazione. Sarà pertanto sempre possibile prevedere mezzi atti a tale discriminazione. Tale discriminazione consente pertanto di immagazzinare i singoli frame di informazione privi della parte di intestazione e comprendenti la sola parte di corpo, contenente cioè le informazioni tipiche della particolare sintassi applicativa utilizzata, ed il messaggio da trasmettere.

I dati incorporanti l'istante temporale di rilevamento e suddivisi in frame di informazione e frame di controllo vengono immagazzinati all'interno di una unità 17 di memorizzazione dei dati rilevati, collegata in maniera bidirezionale all'unità di controllo 15 come rappresentato tramite la freccia F13. E' poi presente una unità 18 di memorizzazione di dati predeterminati, collegata in maniera bidirezionale all'unità di controllo 15. Tali dati predeterminati rappresentano possibili interpretazioni dei frame di informazione o di controllo contenuti nell'unità di memorizzazione 17. Il loro utilizzo

verrà spiegato in seguito con riferimento alle successive figure. Il collegamento tra l'unità di memorizzazione 18 e l'unità di controllo 15 è rappresentato tramite la freccia F14.

Verrà fatto successivamente riferimento alla figura 5, che mostra uno schema di flusso indicante le operazioni che vengono effettuate dall'unità di controllo 15 sui frame di informazione immagazzinati nell'unità 17 di memorizzazione. E' da intendersi che l'accesso a tale frame potrà eventualmente essere selettivamente regolato tramite sistemi di gestione privilegi ed autorizzazioni quali ad esempio password, codici di criptazione, decriptazione, lettori di badge e similari in possesso di utenti abilitati, a seconda dei casi.

Un primo passo S1 indica la lettura dei vari pacchetti avvenuta tramite l'unità 3 di rilevamento. Un secondo passo S2 indica la distinzione, precedentemente descritta, che l'unità di controllo 15 effettua tra i frame di informazione ed i frame di controllo, unitamente all'associazione dell'istante temporale di rilevamento.

Sui frame di controllo, di basso livello (non applicativo), il cui utilizzo è marginale ai fini della presente invenzione, potrà comunque essere

prevista una elaborazione statistica, effettuata nel passo S3. Tale elaborazione non viene qui descritta in dettaglio; le modalità con le quali essa avviene risulteranno chiare in seguito. Il risultato finale di una tale elaborazione fornirà una elencazione dei vari frame di controllo, riportando inoltre il conteggio del numero di occorrenze di ciascuno di tali frame.

Per quanto concerne i frame di informazione, il flusso procede verso un passo S4 in cui i singoli frame di informazione vengono ricostruiti in base alla sintassi applicativa specifica degli stessi. Ai fini di tale ricostruzione, le strutture di sintassi applicativa dei singoli frame di informazione devono essere note. Esse sono infatti contenute all'interno dell'unità 18 di memorizzazione di dati predeterminati descritta con riferimento alla precedente figura 3. Tale unità 18 contiene, ad esempio in un "file" di testo, una descrizione formale astratta di possibili interpretazioni dei frame di informazione o di controllo. Tali dati rappresentano le modalità secondo le quali può essere strutturata la parte di corpo di un singolo frame di informazione, ad esempio il codice di trasmissione macchina (vale a dire relativo ad un frame di informazione spedito dalla sorgente oppure dal destinatario), il numero di canale (vale a dire

relativo ad uno specifico elaboratore a monte del nodo di sorgente oppure ad uno specifico elaboratore a valle del nodo di destinazione), numeri di protocollo, numeri meccanografici etc. E' da intendersi che tale unità 18 può contenere sintassi di molteplici protocolli applicativi, dei frame di informazione da ricostruire in quel momento.

Tramite un confronto sequenziale della parte di corpo di ciascun frame di informazione con ciascuna delle tipologie astratte presenti nell'unità 18, si ottiene una ricostruzione dei singoli frame di informazione.

Successivamente a ciò, si è in grado di ricomporre le varie sequenze applicative intercorse tra un determinato elaboratore di sorgente ed un determinato elaboratore di destinazione, vale a dire un ordinamento temporale e secondo il tipo di comunicazione. Per sequenza applicativa verrà inteso nel corso della presente descrizione l'insieme dei frame di informazione scambiati tra un determinato elaboratore di sorgente ed un determinato elaboratore di destinazione all'interno di una singola comunicazione. La sequenza applicativa ordinata all'interno del passo S5 conterrà i singoli frame di informazione ordinati solamente secondo un criterio

temporale e non anche secondo un criterio logico. L'ordinamento temporale è stato reso possibile dall'associazione dell'istante temporale avvenuta nel precedente passo S2.

Ai fini di un ordinamento anche logico dei dati all'interno di una specifica sequenza applicativa può rivelarsi utile, ma non necessaria, la presenza di un insieme di regole applicative che governano lo scambio di dati tra sorgente e destinazione. Tali regole applicative, tipiche della particolare tipologia di colloquio tra un determinato elaboratore di sorgente ed un determinato elaboratore di destinazione, devono essere predefinite e come tali sono anch'esse raccolte nell'unità 18 di memorizzazione di dati predeterminati. Tali regole applicative sono un insieme di possibili interpretazioni di sequenze di frame di informazione contenuti nell'unità di memorizzazione 17 dei dati rilevati.

Un esempio di regole applicative è dato dalla seguente tabella 1, in cui si fa riferimento ad una comunicazione tra una sorgente che rappresenta uno studente (client) che voglia effettuare una iscrizione via terminale all'università, ed un destinatario (server) che rappresenta l'università cui lo studente vuole iscriversi.

TABELLA 1

1: AS ? FDB 15 AS ? FDB 5 AS ? FDB 0 La prenotazione dell'iscrizione è stata acquisita regolarmente
2: AS ? FDB 13 AS ? FDB 0 La posizione dell'utente non è regolare
.....
.....
.....

Ogni riga di tale tabella è una regola applicativa, indicante cioè una possibile sequenza applicativa di scambio dati tra sorgente e destinatario. Viene qui di seguito riportato il significato di ciascuna di tali sequenze applicative. La prima riga indica ad esempio la seguente sequenza di frame di informazione:

- la sorgente (AS) interroga (?) il destinatario;
- il destinatario (FDB) risponde con l'attività numero 15;
- la sorgente (AS) interroga (?) nuovamente il destinatario;
- il destinatario (FDB) risponde con l'attività numero 5;
- la sorgente (AS) interroga (?) il destinatario; e

- il destinatario (FDB) risponde con l'attività numero 0.

Il risultato cui si perviene al termine di tale conversazione è che la prenotazione dell'iscrizione all'università è stata acquisita regolarmente.

La tabella 1, meramente esemplificativa, potrebbe essere anche rappresentata tramite una struttura ad albero con più o meno ramificazioni, a seconda del numero di sequenze applicative previste. Ogni percorso fino ad una delle foglie dell'albero rappresenterebbe una particolare sequenza applicativa, vale a dire un particolare conversazione tra sorgente e destinatario, vale a dire ancora una particolare sequenza di frame di informazione tra sorgente e destinatario.

Le regole applicative possono essere in numero qualunque. Maggiore sarà il numero di regole applicative fornito, maggiore sarà la possibilità di associare a ciascuna delle sequenze applicative temporalmente ricostruite nel passo S5 un significato logico ben definito, riscontrato tramite confronto con una particolare regola applicativa contenuta nell'unità di memorizzazione 18 di figura 3. In tale modo sarà dunque possibile verificare correttezza o anomalia della particolare sequenza applicativa in quel momento confrontata.

Nel passo S6 di figura 5 l'unità di controllo 15 verifica innanzitutto se tali regole applicative siano disponibili o meno. Supponendo che tali regole applicative siano note, il flusso può procedere o verso un passo S8 oppure verso un passo S9, a seconda di quanto scelto nel passo S7. Il passo S8 permette una semplice classificazione delle sequenze applicative. Ciascuna sequenza applicativa viene infatti classificata come appartenente ad un particolare percorso tra i vari percorsi possibili all'interno dell'albero delle regole applicative. Il passo S8 verrà spiegato in maggiore dettaglio con riferimento alla successiva figura 6.

Nel passo S9 invece, viene ricostruito il percorso logico di tutte le sequenze applicative rilevate dall'apparecchio in un predeterminato intervallo temporale. Tale passo S9 verrà spiegato in maggiore dettaglio con riferimento alla successiva figura 7.

L'apparecchio secondo la presente invenzione consente di effettuare una ricostruzione del percorso logico delle sequenze applicative anche nel caso in cui non sia previsto un insieme di regole applicative. Il flusso procede in tale caso verso un passo S10, anch'esso successivamente descritto.

Verrà fatto ora riferimento alla figura 6, che spiega in maggiore dettaglio quanto sopra descritto con riferimento al passo S8 di figura 5. In un primo passo S11 viene selezionata la singola sequenza applicativa oggetto del confronto. In un successivo passo S12 vengono selezionati, all'interno della sequenza applicativa selezionata, gli elementi caratterizzanti ai fini del confronto.

Nel caso esemplificativo precedentemente descritto di iscrizione all'università con riferimento alla tabella 1 tali elementi caratterizzanti potranno essere: l'identificativo dell'elaboratore di sorgente, l'identificativo dell'utente che ha richiesto l'operazione di iscrizione, i dati forniti dalla sorgente ed i dati forniti dal destinatario.

Nel passo S13 gli elementi caratterizzanti della sequenza applicativa in oggetto vengono confrontati con una delle regole applicative di cui alla precedente tabella 1 alla ricerca di una possibile corrispondenza. Nel caso in cui tale corrispondenza sia stata trovata, il flusso procede verso un passo S14 in cui tale corrispondenza viene segnalata e della quale andrà tenuto conto nei risultati dell'interpretazione. Il flusso torna poi a selezionare una successiva sequenza e a rieseguire il

passo S11. Nel caso in cui la corrispondenza di cui al passo S13 non sia stata trovata, l'unità di controllo 15 passa ad una successiva regola nel passo S15 e nel caso in cui (passo S16) vi siano ancora regole con le quali effettuare il confronto l'unità di controllo ritorna ad eseguire il confronto di cui al passo S13. Nel caso in cui invece non vi siano regole ulteriori, l'unità di controllo segnala una anomalia nel passo S17. Una tale anomalia può alternativamente significare:

- un tipo di sequenza che non sarebbe dovuto avvenire (anomalia vera e propria); oppure
- un tipo di sequenza non inserito per errore all'interno dell'albero delle regole applicative.

In ciascuno di tali casi il riscontro di una tale anomalia è sicuramente utile ai fini della certificazione delle tipologie di sequenze applicative intercorse nel tratto di rete posto sotto osservazione.

Verrà fatto ora riferimento alla successiva figura 6 che spiega in maggiore dettaglio quanto descritto nel passo S9 di figura 5.

I passi S18 ed S19 servono rispettivamente a selezionare la singola sequenza applicativa e gli elementi caratterizzanti della stessa, similmente a

quanto descritto con riferimento alla precedente figura 5. Il passo S20 serve ad indicare il confronto tra la sequenza applicativa e le regole applicative predefinite contenute all'interno dell'unità 18 di memorizzazione di dati predeterminati. Nel caso in cui si sia trovata una corrispondenza, il flusso procede verso un passo S21 in cui viene tenuto conto della corrispondenza rinvenuta tramite aggiornamento dei relativi campi statistici. I passi S18-S20 verranno successivamente ripetuti, fino ad esaurimento delle sequenze da classificare. Nel caso in cui invece non vengano trovate corrispondenze, la sequenza applicativa da classificare è nuova; essa può rappresentare una anomalia oppure semplicemente una sequenza che non è stata prevista. In questo caso il flusso procede verso un passo S22 in cui vengono inizializzati i campi statistici relativi a quella specifica sequenza. La sequenza riscontrata potrà inoltre inserita nella lista delle sequenza predefinite che servono ad effettuare la comparazione nel passo S20. Tale fatto è anche indicato dal doppio senso della freccia F14 della precedente figura 4. E' da intendersi che tali sequenze particolari, le probabili anomalie cioè, possono eventualmente essere marcate in maniera particolare in modo da essere

riconosciute come tali. Successivamente a ciò vengono anche in questo caso ripetuti i passi S18-S20 fino ad esaurimento delle sequenze da classificare. In particolare, oltre a poter individuare il numero di attraversamenti di ciascun ramo dell'albero, sarà possibile individuare anche rami non percorsi.

Nel caso in cui non sia presente una sequenza predefinita di regole applicative, l'unità di controllo sarà sempre in grado di effettuare una ricostruzione delle comunicazioni applicative intercorse sul tratto di rete sotto esame (passo S9 di figura 5). In tale caso ciascuna sequenza applicativa analizzata verrà confrontata non con sequenze predefinite, bensì con le sequenze precedentemente analizzate. L'albero applicativo arricchito di informazioni di tipo statistico verrà pertanto ricostruito tramite confronto reciproco di ciascuna parte di corpo dei frame di informazione con le altre. Verrà anche in questo caso formato un albero e sarà possibile conoscere il numero di attraversamenti di ciascun ramo. In questo caso non sarà ovviamente possibile individuare rami non percorsi, in quanto non si sarà a priori a conoscenza dell'esistenza di tali rami.

Verrà fatto ora riferimento alle figure 8A e 8B

che mostrano rispettivamente una struttura esemplificativa di frame informativo ed una struttura esemplificativa di albero applicativo arricchito di informazioni di tipo statistico ottenuta tramite l'apparecchio secondo la presente invenzione.

In figura 8A è possibile scorgere quattro campi differenti: un primo campo 19 che indica il nominativo dell'elaboratore di sorgente o dell'elaboratore di destinazione; un secondo campo 20 che indica il numero di collegamenti all'interno dell'intervallo di tempo di rilevamento, un terzo campo 21 che indica la durata media di ciascun collegamento, ad esempio in millisecondi, ed un quarto campo 22 che indica il codice dell'attività svolta.

La figura 8B indica l'albero ricostruito. Un primo elemento E1 dell'albero indica che AS (sorgente) si è collegato 20 volte, con una durata media di collegamento di 0 millisecondi (semplice apertura del collegamento con il destinatario) e ha effettuato l'attività con il codice 0. Un secondo elemento E2, unico "figlio" di E1, indica che in tutti e 20 questi collegamenti FDB (destinatario) ha risposto con l'attività con il codice 20, con una durata media di collegamento di 20 millisecondi. Due sono state le modalità con le quali si è proseguito. Per 18 volte

(elemento E3) AS ha risposto con l'attività 0 e per due volte (elemento E4) AS ha risposto con l'attività 1. L'albero prosegue con altri elementi, il cui significato risulta ora chiaro dal contesto. L'albero qui presentato è il risultato dell'ordinamento logico effettuato nei passi S9 o S10 della figura 5.

Si fa notare che l'individuazione dei contenuti del campo 19 e del campo 22 di ciascun elemento è stata effettuata tramite il passo S4 di figura 5. L'individuazione dei collegamenti tra i vari elementi, vale a dire il fatto che l'elemento E2 è "figlio" di E1 e che gli elementi E3 ed E4 sono "figli" di E2 è stata fatta o nel passo S9 oppure nel passo S10 di figura 5.

Terminata la descrizione dettagliata di un apparecchio per il rilevamento e l'interpretazione di protocolli applicativi su rete, verranno qui di seguito descritte in dettaglio la struttura ed il funzionamento delle rimanenti componenti del dispositivo di controllo di accessi secondo la presente invenzione.

Le modalità di connessione preferita di tale dispositivo sono in serie, su reti Ethernet da 10 Mbits (connettori rj58 e rj45) e 100 Mbits (rj45).

I protocolli di livello OSI 2 supportati saranno

tutti i protocolli incapsulati in Ethernet, quali 802.3, DOD IP, ARP etc.

I protocolli di livello OSI 3 supportati saranno tutti i protocolli incapsulati nei vari protocolli di livello OSI 2, quali TCP in IP, UDP in IP, Netbios in IEEE 802.3, SNA in IEEE 802.3 etc.

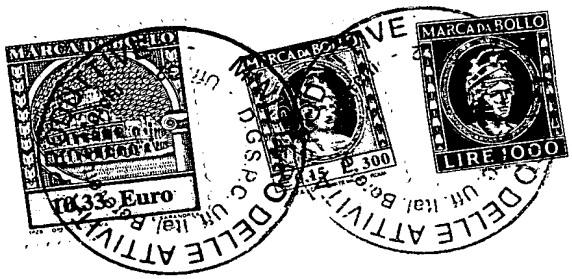
Verrà fatto innanzitutto riferimento alla figura 9, che mostra uno schema a blocchi del dispositivo di controllo di accessi secondo la presente invenzione. I vari blocchi di figura 9 verranno qui di seguito descritti uno alla volta.

Elemento 201:

E' l'elemento di memorizzazione delle regole di riconoscimento dei pattern. L'archivio delle regole di riconoscimento è formato leggendo un file oppure ad esempio digitando direttamente le regole tramite tastiera.

Si supporrà inizialmente che tali regole di riconoscimento si presentano quali coppie <tipo di dato>/<valore dato>.

Un pattern per il riconoscimento di una richiesta di browsing Internet da parte di un client con indirizzo 192.23.40.1 verso un server web di indirizzo 210.20.20.6 presenta ad esempio la seguente struttura: (ETH_PROT, IP),



(IP_SRC_ADDR, 192.23.40.1),

(IP_DST_ADDR, 210.20.20.6),

(TCP_DST_PORT, HTTP)

in cui:

la prima coppia (ETH_PROT, IP) indica che il protocollo contenuto nel livello Ethernet deve essere il protocollo IP;

la seconda coppia (IP_SRC_ADDR, 192.23.40.1) indica che l'indirizzo IP di chi invia il pacchetto deve essere quello indicato;

la terza coppia (IP_DST_ADDR, 210.20.20.6) indica che l'indirizzo IP di chi riceve il pacchetto deve essere quello indicato; e

la quarta coppia (TCP_DST_PORT, HTTP) indica che il servizio TCP utilizzato è quello HTTP (web).

Gli identificativi posti a destra delle coppie possono assumere anche valori non definiti a priori, ad esempio nel caso in cui si vogliano identificare tutti gli indirizzi di una sottorete. In tal caso l'indirizzo dell'esempio precedente può essere espresso come 210.20.20.* dove il simbolo * (asterisco) indica un valore jolly, vale a dire tutti i possibili valori che possono essere presenti in quella posizione. Nell'ambito della stessa coppia possono apparire due o più asterischi: ad esempio

210.*.20.* indicando così un insieme di 65536 (o più) indirizzi diversi. Altre forme accettate sono ad esempio: 2*.20.20.* indicando tutti gli indirizzi che iniziano con 2 e terminano con un sottoindirizzo tra 0 e 255 (in totale in questo caso 100*256=25600 indirizzi diversi).

Un ulteriore esempio di pattern per il riconoscimento del protocollo IBM NetBios tra due elaboratori è il seguente:

(ETH_PROT, IEEE802),

(IEEE802_DST_SAP, IBM_NETBIOS)

Volendo inoltre forzare anche il riconoscimento delle schede di rete (sono 6 bytes che includono il codice costruttore della scheda ed il numero della scheda) coinvolte nella comunicazione NetBios, il pattern diventa:

(ETH_SRC_ADDR, 0xFF45DE782201),

(ETH_DST_ADDR, 0xF237C811000F),

(ETH_PROT, IEEE802),

(IEEE802_DST_SAP, IBM_NETBIOS).

Elemento 202:

E' il compilatore di pattern ed è composto da un elemento di conversione delle regole contenute in 201 in un insieme di sequenze di identificatori numerici e da un elemento di compressione degli identificatori

così ottenuti.

i) Elemento di conversione

Le regole di riconoscimento che si presentano quali coppie <tipo di dato>/<valore dato> vengono convertite in sequenze di identificatori numerici che costituiscono la base di riconoscimento delle trame lette da rete.

Data ad esempio la regola

(ETH_PROT, IP),

(IP_SRC_ADDR, 228.186.33.90),

(IP_DST_ADDR, 41.240.227.149),

(TCP_DST_PORT, HTTP)

abbiamo che:

a) la prima coppia (ETH_PROT, IP) viene convertita in due coppie di dati in esadecimale (in cui il prefisso 0x indica che il valore successivo è rappresentato in esadecimale):

0x0C 0x0800

0x49 0x06

in cui:

- la prima riga contiene due valori, 0C e 0800. La cifra più a sinistra del primo valore (0) indica che si è in presenza di una trama Ethernet. La seconda cifra del primo valore (C) indica la posizione all'interno della trama (13° byte, considerando il

primo in posizione 0). Il secondo valore (0800) è il codice identificativo del protocollo IP quando contenuto in una trama Ehternet; e

- la seconda riga contiene due valori, 49 e 06. La cifra più a sinistra del primo valore (4) indica che si è in presenza di una trama IP. La seconda cifra del primo valore (9) indica la posizione all'interno della trama. Il secondo valore (06) è identificativo del protocollo TCP contenuto in IP.

b) la seconda coppia (IP_SRC_ADDR, 228.186.33.90) viene convertita in quattro coppie di dati in esadecimale:

0x4C 0xe4

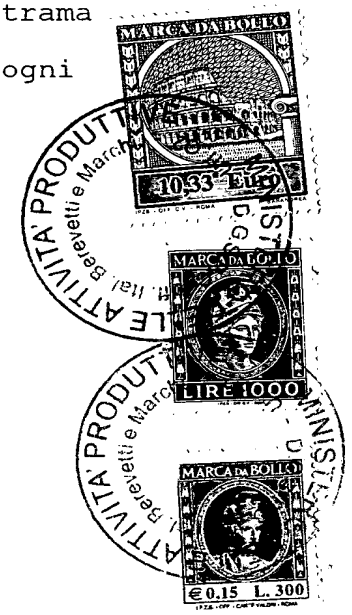
0x4D 0xba

0x4E 0x21

0x4F 0x5a

in cui ciascuna coppia indica rispettivamente la trama IP (4), la posizione (da C ad F) ed il valore di ogni singolo elemento che costituisce l'indirizzo del trasmettitore (source address): e4 in esadecimale corrisponde infatti a 228 in decimale, ba in esadecimale corrisponde a 186 in decimale, 21 in esadecimale corrisponde a 33 in decimale e 5a in esadecimale corrisponde a 90 in decimale.

c) la terza coppia (IP_DST_ADDR, 41.240.227.149)



viene convertita in quattro coppie di dati in esadecimale:

0x410 0x29

0x411 0xF0

0x412 0xE3

0x413 0x95

in cui ciascuna coppia indica rispettivamente la trama IP (4), la posizione (da 10 a 13) ed il valore di ogni singolo elemento che costituisce l'indirizzo del ricevitore (destination address): 29 in esadecimale corrisponde infatti a 41 in decimale, F0 in esadecimale corrisponde a 240 in decimale, E3 in esadecimale corrisponde a 227 in decimale e 95 in esadecimale corrisponde a 149 in decimale.

d) la quarta coppia (TCP_DST_PORT, HTTP) viene convertita in una coppia di dati in esadecimale:

0x82 0x0080

in cui la cifra più a sinistra del primo valore (8) indica la trama TCP, la seconda cifra del primo valore (2) indica la posizione all'interno di tale trama (la terza, partendo da zero) mentre il secondo valore 0080 indica il servizio HTTP (quello utilizzato dalle applicazioni web).

Quindi, a partire dalla regola

(ETH_PROT, IP),

(IP_SRC_ADDR, 228.186.33.90),

(IP_DST_ADDR, 41.240.227.149),

(TCP_DST_PORT, HTTP)

si ottiene la sequenza

0x0C 0x0800, 0x49 0x06, 0x4C 0xe4, 0x4D 0xba, 0x4E 0x21, 0x4F 0x5a, 0x410 0x29, 0x411 0xf0, 0x412 0xe3, 0x413 0x95, 0x82 0x0080.

E' da intendersi che tutte le conversioni fin qui descritte sono rese possibili tramite un confronto sequenziale di ciascuna delle coppie <tipo di dato>/<valore dato> con una tabella memorizzante tutte le possibili coppie <tipo di dato>/<valore dato> unitamente alla corrispondente coppia di valori esadecimali.

In realtà per le regole così definite può utilizzarsi un formalismo più esteso, semanticamente rappresentabile dalla coppia <oggetto>/<azione>. Il campo <oggetto> indica l'insieme di proprietà (compreso il valore) assunte dall'elemento correntemente sotto esame, mentre il campo <azione> esprime le azioni da eseguire dopo aver riconosciuto tale oggetto nella trama di comunicazione.

Nelle coppie esadecimali del tipo <tipo di dato>/<valore dato> è ad esempio facile notare come il campo <tipo di dato> contenga una doppia informazione,

vale a dire sia il protocollo (o il tipo di trama) cui si fa riferimento, sia anche la posizione all'interno di tale protocollo.

Nel caso di protocolli applicativi complessi, le trame rilevate vengono in genere rappresentate tramite un linguaggio di tipo LL(1) (vale a dire, secondo la definizione di Chomsky, un linguaggio che non presenta strutture di controllo e che non presenta limiti per la definizione dei meccanismi di interpretazione delle strutture informative). In tale caso il campo <azione> farà riferimento ad un insieme minimo di comandi elementari, qui riportati:

- **Push**

<valore>

<variabile>

<posizione di lettura>

<valore alla posizione di lettura>

- **Pop**

<variabile>

<posizione di lettura>

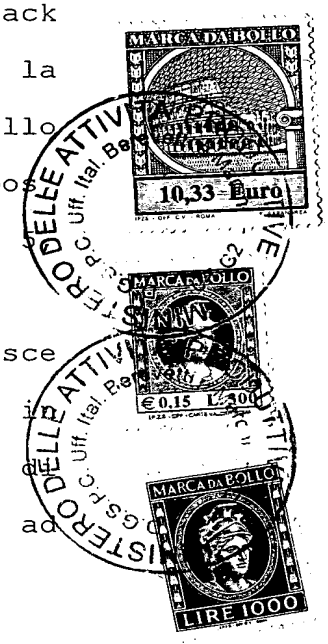
<nella posizione di lettura>

- **And**
- **Mul**
- **Add**
- **Equal**

- **Next**
- **F_send_all**
- **F_dynamic**

Qui di seguito viene fornito, a scopo di completezza, un breve cenno al significato di tali comandi elementari.

- Push <valore> inserisce un valore nello stack riservato al processo di riconoscimento in atto, ad esempio: PUSH(35), il valore 35 viene messo nello stack;
- Push <variabile> inserisce il contenuto di una variabile nello stack riservato al processo di riconoscimento in atto, ad esempio:PUSH(v12), se la variabile "v12" vale 8, 8 viene messo nello stack;
- Push <posizione di lettura> inserisce nello stack riservato al processo di riconoscimento in atto la posizione del valore correntemente da leggere nello stream di input, ad esempio PUSH(pos) se pos variabile che indica la posizione di lettura, vale allora 5 viene messo nello stack;
- Push <valore alla posizione di lettura> inserisce nello stack riservato al processo di riconoscimento in atto il valore letto nello stream di input in fase di riconoscimento al posto "posizione di lettura", ad



esempio PUSH(v_pos), se pos, variabile che indica la posizione di lettura, vale 5 e se al posto 5 dello stream di input c'è il valore 30, allora 30 viene messo nello stack;

- Pop <variabile> inserisce la testa dello stack nella variabile "variabile", ad esempio POP(v3) ,se nella testa dello stack è stato inserito il valore 10, significa che l'ultima operazione fatta con lo stack è stata ad esempio push(10), il valore 10 va nella variabile "v3";
- Pop <posizione di lettura> inserisce la testa dello stack nella variabile che indica la prossima posizione da leggere nello stream di input, ad esempio POP(pos) ,se nella testa dello stack è stato inserito il valore 10, il prossimo elemento che verrà letto dallo stream di input sarà quello in posizione 10;
- Pop <nella posizione di lettura> inserisce la testa dello stack nella posizione indicata dalla variabile che indica la prossima posizione da leggere nello stream di input, ad esempio POP(v_pos) ,se nella testa dello stack è stato inserito il valore 10, il prossimo elemento che verrà letto dallo stream di input avrà valore 10;
- And, Mul, Add, Or, Sub sono tutte operazioni aritmetiche e logiche. L'operazione viene svolta sui

valori contenuti nelle prime due posizione dello stack, il risultato diventa la testa dello stack e i due valori utilizzati sono tolti dallo stack; esempio: le operazioni logico aritmetiche seguono la notazione polacca inversa (RPN). Supponiamo di dover eseguire l'operazione 10*30: il programma che ne consegue avrà la seguente forma:

PUSH(10)

PUSH(30)

MUL

ora, nella testa dello stack c'è 300=30*10.

- Equal <valore>, Equal <variabile>, Equal <posizione di lettura>, Equal <valore nella posizione di lettura> verifica se nella testa dello stack c'è un valore uguale a quello passato come parametro. Il risultato (0 se diversi, 1 se uguali) viene messo nella testa dello stack;
- f_send_all è una funzione che, quando eseguita, riporta verso l'output l'intero stream di input;
- Next <valore>, Next <variabile> incrementa di "valore" oppure del valore contenuto nella "variabile" la variabile che indica la posizione nello stream di input dove leggere il prossimo valore; ed infine
- f_dynamic("nome") esegue la funzione "nome" collegata all'elemento di coordinamento attraverso

meccanismi di collegamento dinamico (tipo DLL di windows oppure shared_libraries di UNIX oppure meccanismi RPC di DCE, ...) passandogli come parametri i valori contenuti nello stack.

Una possibile sintassi implementativa (ripresa dal linguaggio C) dell'insieme delle coppie <oggetto>/<azione> potrà essere la seguente:

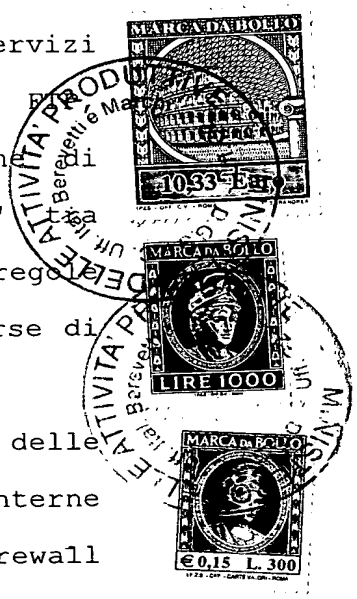
```
typedef struct _item {
        unsigned char oggetto;
        unsigned long int azione;
} Item;

typedef struct _record {
     int num_of_items;
     Item * items;
} Record;

Record * input_seconda_fase;
```

in cui:

- il campo "oggetto" è stato espresso come un singolo byte ("unsigned char"). Una tale scelta non comporta limitazioni in quanto un valore intero (lungo da 2 a 4 byte) può essere considerato come una sequenza di byte e quindi può essere trattato un byte alla volta; e

- il campo "azione" è stato espresso come "unsigned long int". Può quindi rappresentare sia un numero (compatibile con la prima notazione) sia come puntatore ad una struttura o insieme di funzioni (compatibile con la seconda notazione).

Il numero di sequenze diverse è in genere molto elevato. A titolo di esempio, considerando esclusivamente il protocollo TCP-IP, per un numero relativamente esiguo di 1000 "clients" (cioè di elaboratori che utilizzano servizi applicativi resi disponibili da altri elaboratori) e di 10 "servers" (cioè di elaboratori che forniscono i servizi applicativi ai clients) e di una media di 10 servizi applicativi per "server" (quali ad esempio FTP, TELNET, HTTP, MAIL, NFS, TIME, DNS), al fine di distinguere tutti i possibili "accoppiamenti" tra client-server-servizio, è necessario definire regole atte ad indicare 1000*10*10=100000 sequenze diverse di pattern nelle trame di comunicazione.



Questo numero, già ben al di sopra delle dimensioni ritenute accettabili per le tavole interne di indirizzamento dei router e dei firewall commerciali, cresce in modo molto rapido quando si vanno a definire regole che agiscono non solo a livello di parte di controllo dei protocolli di

comunicazione ma anche a livello di parte dati, secondo quanto accade nella presente invenzione.

Il linguaggio delle regole sopra definito consente di redigere regole che permettono di identificare elementi della parte dati dei protocollo di comunicazione: se si vuole infatti non soltanto "identificare" un "client" ma anche quando questo su rete cerca di accedere ad una particolare pagina WEB (cosa del tutto possibile tramite la presente invenzione), non è sufficiente agire a livello di parte di controllo del protocollo di comunicazione (si riuscirebbe a riconoscere solo il fatto che è stato inviato un comando a livello di servizio HTTP) ma è necessario agire a livello di parte dati del protocollo TCP-IP al fine di identificare la particolare stringa che determina l'accesso alla pagina WEB richiesta dal client.

ii) Elemento di compressione dell'insieme di sequenze ottenute in una struttura dati ad accesso diretto

Tale secondo elemento del compilatore di pattern 202 permette una costruzione della struttura dati di compressione che è in grado di garantire un tempo di accesso costante (indipendente cioè dalla numerosità delle sequenze) ed un'occupazione di memoria ottimale

(pari cioè, a meno di una costante moltiplicativa, alla quantità di memoria necessaria per memorizzare le sequenze in modo non strutturato) per il riconoscimento delle sequenze memorizzate in tale struttura nelle trame di comunicazione che si leggono da rete.

Verrà in particolare fatto riferimento agli articoli:

a) "Time Optimal Digraph Browsing on a Sparse Representation", Tech. Report, Dipartimento di Matematica, Università di Roma "Tor Vergata", 8/97, 1997 di M. Talamo e P. Vocca;

b) "Optimal Bounds on Complexity of Sparse Partial Orders", Tech. Report Dipartimento di Matematica, Università di Roma "Tor Vergata", 9/97, 1997 di M. Talamo e P. Vocca;

c) "Optimal Digraph Search on a Compressed Representation", Tech. Report Dipartimento di Matematica, Università di Roma "Tor Vergata", 11/98, 1998 di M. Talamo e P. Vocca; e

d) "Compact Implicit Representation of Graphs", proceedings WG98, giugno 1998 di M. Talamo e P. Vocca.

In tali scritti si rappresentano strutture dati che permettono una accessibilità a tempo costante, vale a dire indipendente dalla numerosità di dati che

queste rappresentano.

L'algoritmo per l'ottenimento di tali strutture di dati si applica a strutture di ingresso del tipo a "grafo bipartito", come ad esempio rappresentato in figura 10A. In tale grafo i nodi possono essere distinti in due sottoinsiemi distinti (da A ad E e da 0 a 4 in figura), in maniera tale che ciascun nodo appartenente ad un primo sottoinsieme possa essere connesso solamente a nodi appartenenti al secondo sottoinsieme e viceversa. Con riferimento alla figura 10A, il nodo A è connesso al nodo 0 ed al nodo 2, il nodo B è connesso al nodo 0 ed al nodo 2, il nodo C è connesso al nodo 1 ed al nodo 4, il nodo D è connesso al nodo 3 ed il nodo E è connesso al nodo 3.

Tali connessioni possono essere espresse mediante una matrice bidimensionale del tipo riportato in figura 10B, ove con il simbolo x sono state riportate le connessioni attive tra righe e colonne. Si può pertanto concludere che i grafi bipartiti sono equivalenti alle matrici bidimensionali e che pertanto i risultati di accessibilità a tempo costante ottenuti con riferimento all'articolo sopra citato possono applicarsi anche a strutture quali matrici bidimensionali.

L'elemento di compressione sarà pertanto tale da

comprimere le sequenze ottenute tramite l'elemento di conversione e da generare una molteplicità di matrici bidimensionali indicative di tali sequenze.

* * *

L'algoritmo tramite il quale l'elemento di compressione opera, qui di seguito presentato, (da PASSO 1 a PASSO 11) è da intendersi implementato in un qualsiasi linguaggio di programmazione adatto allo scopo (ad esempio in linguaggio C) e memorizzato all'interno di una ROM.

L'input dell'algoritmo consiste nella sequenza di identificativi numerici (record) a lunghezza variabile precedentemente specificata.

Congiuntamente ai vari passi dell'algoritmo, si riporterà a livello esemplificativo un ciclo completo di compilazione per un particolare caso pratico, al fine di descrivere in maniera compiuta le modalità operative dell'algoritmo stesso. A scopo esemplificativo si continuerà a fare riferimento a strutture di comunicazione del tipo Ethernet. E' da intendersi che il funzionamento del dispositivo di controllo secondo la presente invenzione resta inalterato anche nel caso in cui l'apparecchio di rilevamento ed interpretazione non fornisca trame Ethernet rilevate sulla rete ma fornisca direttamente

comunicazioni a livello TCP/IP o comunque stream di dati anche molto lunghi.

PASSO 1 (specifica regole predefinite, si veda anche la figura 11):

Si supponga di dover gestire e coordinare trame di comunicazione a livello Ethernet tramite i seguenti schemi di connettività:

connessione a) 132.147.200.10 potrà connettersi a 132.147.160.1 solamente per il servizio:

- WWW servizio TCP 80.

connessione b) 132.147.200.10 potrà connettersi a 132.147.160.2 solamente per i servizi:

- SMTP servizio TCP 25;
- NETBIOS servizi TCP 137, 138 e 139.

connessione c) 132.147.200.20 potrà connettersi a 132.147.160.1 solamente per i servizi:

- FTP servizio TCP 20 e 21;
- TELNET servizio TCP 23.

connessione d) 132.147.200.20 potrà connettersi a 132.147.160.2 solamente per i servizi:

- SMTP servizio TCP 25;
- WWW servizio TCP 80.

connessione e) 132.147.200.20 potrà connettersi a 132.147.160.3 solamente per i servizi:

- WWW servizio TCP 80;

- SNMP servizi TCP 161 e 162;
- NFS servizio TCP 2049;
- TELNET servizio TCP 23.

Inoltre dovranno essere ammesse tutte le comunicazioni di tipo ARP (protocollo a livello Ethernet) e ICMP (protocollo a livello IP).

PASSO 2 (conversione regole in un insieme di sequenze):

In base a tale schema di connessione si ottiene un insieme di 17 record (in cui ogni record è formato da un insieme di coppie <oggetto>/<azione>). In particolare, il record 1 rappresenta la connessione a), i record da 2 a 5 rappresentano la connessione b), i record da 6 a 8 rappresentano la connessione c), i record da 9 a 10 rappresentano la connessione d), i record da 11 a 15 rappresentano la connessione e), il record 16 rappresenta il protocollo ARP in Ethernet ed infine il record 17 rappresenta il protocollo ICMP in IP.

Connessione a)

RECORD 1

0x08, 0x000C

0x00, 0x000D protocollo IP in Ethernet

0x06, 0x4009	protocollo TCP in IP
0x84, 0x400C	
0x93, 0x400D	
0xC8, 0x400E	
0x0A, 0x400F	132.147.200.10
0x84, 0x4010	
0x93, 0x4011	
0xA0, 0x4012	
0x01, 0x4013	132.147.160.1
0x00, 0x8002	
0x50, 0x8003	WWW 80

Connessione b)

RECORD 2

0x08, 0x000C	
0x00, 0x000D	protocollo IP in Ethernet
0x06, 0x4009	protocollo TCP in IP
0x84, 0x400C	
0x93, 0x400D	
0xC8, 0x400E	
0x0A, 0x400F	132.147.200.10
0x84, 0x4010	
0x93, 0x4011	
0xA0, 0x4012	
0x02, 0x4013	132.147.160.2

0x00, 0x8002

0x19, 0x8003 SMTP 25

RECORD 3

0x08, 0x000C

0x00, 0x000D protocollo IP in Ethernet

0x06, 0x4009 protocollo TCP in IP

0x84, 0x400C

0x93, 0x400D

0xC8, 0x400E

0x0A, 0x400F 132.147.200.10

0x84, 0x4010

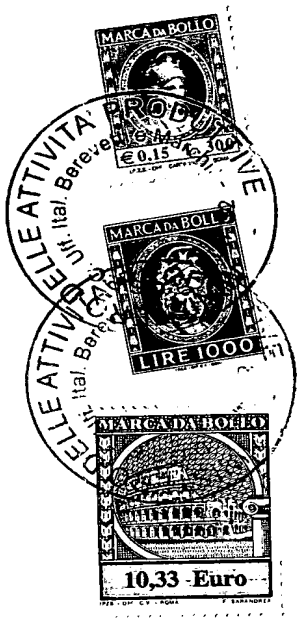
0x93, 0x4011

0xA0, 0x4012

0x02, 0x4013 132.147.160.2

0x00, 0x8002

0x89, 0x8003 NETBIOS 137



RECORD 4

0x08, 0x000C

0x00, 0x000D protocollo IP in Ethernet

0x06, 0x4009 protocollo TCP in IP

0x84, 0x400C

0x93, 0x400D

0xC8, 0x400E

0x0A, 0x400F	132.147.200.10
0x84, 0x4010	
0x93, 0x4011	
0xA0, 0x4012	
0x02, 0x4013	132.147.160.2
0x00, 0x8002	
0x8A, 0x8003	NETBIOS 138

RECORD 5

0x08, 0x000C	
0x00, 0x000D	protocollo IP in Ethernet
0x06, 0x4009	protocollo TCP in IP
0x84, 0x400C	
0x93, 0x400D	
0xC8, 0x400E	
0x0A, 0x400F	132.147.200.10
0x84, 0x4010	
0x93, 0x4011	
0xA0, 0x4012	
0x02, 0x4013	132.147.160.2
0x00, 0x8002	
0x8B, 0x8003	NETBIOS 139

Connessione c)

RECORD 6

```
0x08, 0x000C
0x00, 0x000D        protocollo IP in Ethernet
0x06, 0x4009        protocollo TCP in IP
0x84, 0x400C
0x93, 0x400D
0xC8, 0x400E
0x14, 0x400F        132.147.200.20
0x84, 0x4010
0x93, 0x4011
0xA0, 0x4012
0x01, 0x4013        132.147.160.1
0x00, 0x8002
0x14, 0x8003        FTP 20
```

RECORD 7

```
0x08, 0x000C
0x00, 0x000D        protocollo IP in Ethernet
0x06, 0x4009        protocollo TCP in IP
0x84, 0x400C
0x93, 0x400D
0xC8, 0x400E
0x14, 0x400F        132.147.200.20
0x84, 0x4010
0x93, 0x4011
0xA0, 0x4012
```

0x01, 0x4013 132.147.160.1

0x00, 0x8002

0x15, 0x8003 FTP 21

RECORD 8

0x08, 0x000C

0x00, 0x000D protocollo IP in Ethernet

0x06, 0x4009 protocollo TCP in IP

0x84, 0x400C

0x93, 0x400D

0xC8, 0x400E

0x14, 0x400F 132.147.200.20

0x84, 0x4010

0x93, 0x4011

0xA0, 0x4012

0x01, 0x4013 132.147.160.1

0x00, 0x8002

0x17, 0x8003 TELNET 23

Connessione d)

RECORD 9

0x08, 0x000C

0x00, 0x000D protocollo IP in Ethernet

0x06, 0x4009 protocollo TCP in IP

0x84, 0x400C

```
0x93, 0x400D
0xC8, 0x400E
0x14, 0x400F        132.147.200.20
0x84, 0x4010
0x93, 0x4011
0xA0, 0x4012
0x02, 0x4013        132.147.160.2
0x00, 0x8002
0x19, 0x8003        SMTP 25
```

RECORD 10

```
0x08, 0x000C
0x00, 0x000D        protocollo IP in Ethernet
0x06, 0x4009        protocollo TCP in IP
0x84, 0x400C
0x93, 0x400D
0xC8, 0x400E
0x14, 0x400F        132.147.200.20
0x84, 0x4010
0x93, 0x4011
0xA0, 0x4012
0x02, 0x4013        132.147.160.2
0x00, 0x8002
0x50, 0x8003        WWW 80
```

Connessione e)

RECORD 11

0x08, 0x000C	
0x00, 0x000D	protocollo IP in Ethernet
0x06, 0x4009	protocollo TCP in IP
0x84, 0x400C	
0x93, 0x400D	
0xC8, 0x400E	
0x14, 0x400F	132.147.200.20
0x84, 0x4010	
0x93, 0x4011	
0xA0, 0x4012	
0x03, 0x4013	132.147.160.3
0x00, 0x8002	
0x50, 0x8003	WWW 80

RECORD 12

0x08, 0x000C	
0x00, 0x000D	protocollo IP in Ethernet
0x06, 0x4009	protocollo TCP in IP
0x84, 0x400C	
0x93, 0x400D	
0xC8, 0x400E	
0x14, 0x400F	132.147.200.20
0x84, 0x4010	

```
0x93, 0x4011
0xA0, 0x4012
0x03, 0x4013          132.147.160.3
0x00, 0x8002
0xA1, 0x8003          SNMP 161
```

RECORD 13

```
0x08, 0x000C
0x00, 0x000D          protocollo IP in Ethernet
0x06, 0x4009          protocollo TCP in IP
0x84, 0x400C
0x93, 0x400D
0xC8, 0x400E
0x14, 0x400F          132.147.200.20
0x84, 0x4010
0x93, 0x4011
0xA0, 0x4012
0x03, 0x4013          132.147.160.3
0x00, 0x8002
0xA2, 0x8003          SNMP 162
```

RECORD 14

```
0x08, 0x000C
0x00, 0x000D          protocollo IP in Ethernet
0x06, 0x4009          protocollo TCP in IP
```

```
0x84, 0x400C
0x93, 0x400D
0xC8, 0x400E
0x14, 0x400F        132.147.200.20
0x84, 0x4010
0x93, 0x4011
0xA0, 0x4012
0x03, 0x4013        132.147.160.3
0x08, 0x8002
0x01, 0x8003        NFS 2049
```

RECORD 15

```
0x08, 0x000C
0x00, 0x000D        protocollo IP in Ethernet
0x06, 0x4009        protocollo TCP in IP
0x84, 0x400C
0x93, 0x400D
0xC8, 0x400E
0x14, 0x400F        132.147.200.20
0x84, 0x4010
0x93, 0x4011
0xA0, 0x4012
0x03, 0x4013        132.147.160.3
0x00, 0x8002
0x17, 0x8003        TELNET 23
```

ed infine

RECORD 16

0x08, 0x000C

0x06, 0x000D protocollo ARP in Ethernet

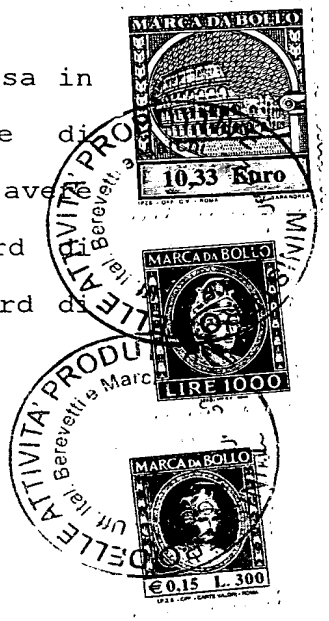
RECORD 17

0x08, 0x000C

0x00, 0x000D protocollo IP in Ethernet

0x01, 0x4009 protocollo ICMP in IP

La struttura così ottenuta può essere espressa in forma matriciale, secondo la rappresentazione di figura 12. Si noti come i vari record possono avere lunghezze diverse. Si hanno infatti 15 record di lunghezza 13, 1 record di lunghezza 2 ed 1 record di lunghezza 3.



PASSO 3:

Si pone CONT = 0

PASSO 4:

Si prende la colonna 0 e la colonna CONT della sequenza sopra riportata e si costruisce una nuova sequenza di record che contengono solo 2 item (quello in colonna 0 e quello in colonna CONT).

PASSO 5:

Da questa nuova sequenza di record si eliminano i duplicati.

PASSO 6:

Si pone RIGA=0

PASSO 7:

Con la nuova sequenza di record si costruisce un grafo bipartito pesato inserendo per ogni record:

- il valore dell'item in posizione 0 (id del nodo superiore);
- il valore dell'item in posizione CONT (id del nodo inferiore);
- l'azione dell'item in posizione CONT (quale primo peso dell'arco tra i due nodi);
- RIGA (quale secondo peso dell'arco tra i due nodi).

Inoltre, per ogni coppia di nodi inserita, si sostituisce al valore dell'item in posizione 0 nella sequenza di record originale il nuovo valore RIGA e si pone RIGA=RIGA+1.

PASSO 8:

Si converte il grafo bipartito così ottenuto in una

matrice bidimensionale ed in un vettore tramite utilizzo dell'algoritmo di base di cui alle pubblicazioni sopra menzionate. Si noti comunque che l'algoritmo qui descritto costituisce un'estensione di tale algoritmo di base, in particolare per quanto riguarda il precedente passo 7.

PASSO 9:

Si memorizzano la matrice bidimensionale ed il vettore.

PASSO 10:

Si pone CONT=CONT+1

PASSO 11:

Se CONT non è pari al numero massimo di item dei record, si torna al passo 4, altrimenti l'algoritmo è terminato.

* * *

La sequenza di matrici bidimensionali e vettori costituisce la struttura dati compressa che sarà utilizzata per il riconoscimento degli stream di ingresso. Tale struttura è accessibile in maniera diretta.

Si farà d'ora in poi nuovamente riferimento alla

figura 9.

Elemento 203 (Memoria contenente i pattern compressi):

Tale elemento è costituito dalla sequenza di matrici ottenute come risultato dell'algoritmo di compressione di cui sopra. Grazie all'elevato grado di compressione di tale algoritmo, la dimensione di questa sequenza di matrici è direttamente proporzionale al numero di connessioni attive della matrice originale e quindi è direttamente memorizzabile in memoria centrale. In caso di elevata numerosità di connessioni attive (>100.000.000) è possibile gestire tale sequenza di matrici compresse tramite dispositivi di memoria di massa.

Elemento 204 (Riconoscitore dei pattern):

Tale elemento permette il confronto tra le trame applicative da riconoscere rilevate tramite l'elemento 205 e la struttura dati ad accesso diretto memorizzata in 203.

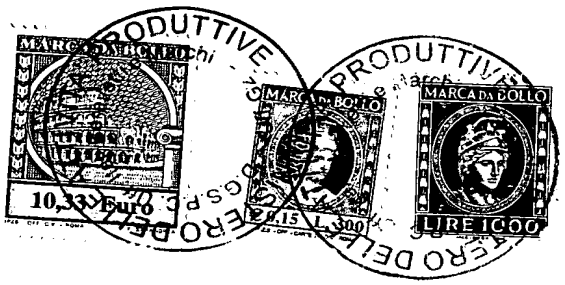
L'elemento 204 è realizzato in un microchip ed è sostanzialmente costituito da un software che implementa una tecnica di accesso diretto su matrici, al fine di accedere alle matrici memorizzate in 203.

Si è pertanto in grado di riconoscere in modo completamente deterministico l'accettabilità o la non accettabilità della trama letta da rete.

* * *

Al fine di fornire un esempio dettagliato del funzionamento di tale riconoscitore, viene qui di seguito innanzitutto riportata la struttura delle matrici memorizzate in 203, tramite utilizzo di una sintassi simile a quella del linguaggio C:

```
//Struttura per una Matrice bidimensionale ed un vettore
typedef struct _matrici_AB {
     unsigned long int row_a;   //Numero di righe della matrice
     unsigned long int col_a;   //Numero di colonne della matrice
     unsigned long int col_b;   //Numero di elementi del vettore
     unsigned long int **mA;    //Matrice dei valori
     Azione ***mP;              //Matrice delle Azioni
     unsigned long int *mB;     //Vettore
} mat_AB;

typedef struct _vec_matrici_AB {
     mat_AB * MAB;          //Insieme delle matrici e dei vettori
     unsigned long int num_mab; //Numero delle matrici e dei vettori
```



```
} * Vec_mat_AB;
```

Si riportano poi qui di seguito cinque record di ingresso e le matrici risultanti. In tale esempio la descrizione dei record viene fatta tramite la sintassi <oggetto>/<azione> precedentemente riportata. Le azioni associate sono estremamente semplificate (una sola azione per ogni riconoscimento). Per semplicità si suppone inoltre che il riconoscimento inizi sempre dal primo byte dello stream in ingresso.

RECORD 1

0x01 next(1)

0x03 next(1)

0x02 f_send_all

RECORD 2

0x01 next(1)

0x06 next(1)

0x04 f_send_all

RECORD 3

0x02 next(1)

0x07 next(1)

0x03 f_send_all

RECORD 4

0x01 next(1)

0x02 f_send_all

RECORD 5

0x05 next(1)

0x01 f_send_all

Per next(1) si intende l'azione di posizionamento sul prossimo byte nello stream di dati. Per f_send_all si intende l'azione di inoltro verso l'esterno di tutto lo stream di dati.

Tramite l'algoritmo precedentemente descritto si viene ad ottenere la seguente struttura matriciale:

1	MATRICE DEI VALORI 0: [X] 1: [0] 2: [1] 3: [X] 4: [X] 5: [2]	MATRICE DELLE AZIONI (1, 0) NEXT(1) (2, 0) NEXT(1) (5, 0) NEXT(1)	VETTORE B X 0 0 X X 0
2	MATRICE DEI VALORI 0: [X]	MATRICE DELLE AZIONI (1, 0)	VETTORE B 0 0 0

	1: [4] 2: [0] 3: [1] 4: [X] 5: [X] 6: [2] 7: [3]	F_SEND_ALL (2, 0) F_SEND_ALL (3, 0) NEXT(1) (6, 0) NEXT(1) (7, 0) NEXT(1)	
3	MATRICE DEI VALORI 0: [0 4] 1: [X X] 2: [1 X] 3: [3 X] 4: [2 X]	MATRICE DELLE AZIONI (2, 0) F_SEND_ALL (3, 0) F_SEND_ALL (4, 0) F_SEND_ALL	VETTORE B 0 0 0 0 1

Per non appesantire la presente descrizione non verranno qui descritte nel dettaglio le varie operazioni (in fondo semplici applicazioni dell'algoritmo sopra descritto) che portano dai record alla struttura matriciale ora graficata. A scopo di chiarezza, la matrice dei valori è stata poi rappresentata fisicamente separata dalla matrice delle

azioni.

Verranno invece descritte in dettaglio le operazioni di confronto che vengono eseguite al fine di riconoscere o meno gli stream di dati rilevati. Tali operazioni si riferiscono al caso particolare della struttura matriciale di cui sopra.

1) ESEMPIO DI RICONOSCIMENTO NEL CASO IN CUI LO STREAM SIA IDENTICO AL RECORD 1: 0x01 0x03 0x02

Il primo valore letto è 01.

Essendo nella condizione iniziale lo si usa sia come indice della matrice che del vettore.

La coppia Matrice/Vettore che si utilizza è nella posizione 1 dell'elenco sopra riportato.

L'indice di riga della matrice A viene determinato dall'elemento letto, vale a dire Riga A =01, cioè la prima riga.

L'indice di colonna della matrice A viene determinato dal valore contenuto dal vettore B nella posizione corrispondente all'elemento letto, vale a dire Colonna A =B[0x01]=0, cioè la zeresima colonna.

Si andrà pertanto a leggere il valore riportato in A[1, 0] e cioè 0. Tale valore costituisce il prossimo indice del vettore B.

Si andrà poi a leggere l'azione riportata in A[1, 0] vale a dire il valore numerico corrispondente

all'azione next(1).

Verrà pertanto eseguita la suddetta azione e dunque ci si posizionerà sul prossimo valore dello stream di dati.

Si andrà al prossimo valore e si utilizzerà quindi la coppia Matrice/Vettore che è nella posizione 2 dell'elenco sopra riportato.

Il valore letto è 03.

L'indice di riga della matrice A viene determinato dall'elemento letto, vale a dire Riga A =03, cioè la terza riga.

L'indice di colonna della matrice A viene determinato dal valore contenuto dal vettore B nella posizione corrispondente al valore riportato in A[1, 0] ottenuto nel passo precedente (e cioè 0). Colonna A =B[0]=0, cioè la zeresima colonna.

Si andrà pertanto a leggere il valore riportato in A[3, 0] e cioè 1. Tale valore costituisce il prossimo indice del vettore B.

Si andrà poi a leggere l'azione riportata in A[3, 0] vale a dire il valore numerico corrispondente all'azione next(1).

Verrà pertanto eseguita la suddetta azione e dunque ci si posizionerà sul prossimo valore dello stream di dati.

Si andrà al prossimo valore e si utilizzerà quindi la coppia Matrice/Vettore che è nella posizione 3 dell'elenco sopra riportato.

Il valore letto è 02.

L'indice di riga della matrice A viene determinato dall'elemento letto, vale a dire Riga A =02, cioè la seconda riga.

L'indice di colonna della matrice A viene determinato dal valore contenuto dal vettore B nella posizione corrispondente al valore riportato in A[3, 0] ottenuto nel passo precedente (e cioè 1). Colonna A =B[1]=0, cioè la zeresima colonna.

Si andrà pertanto a leggere il valore riportato in A[2, 0] e cioè 1. Tale valore costituisce il prossimo indice del vettore B.

Si andrà poi a leggere l'azione riportata in A[2, 0] vale a dire il valore numerico corrispondente all'azione f_send_all. Ciò significa che è avvenuto il riconoscimento.

2) ESEMPIO DI RICONOSCIMENTO NEL CASO IN CUI LO STREAM SIA DIVERSO DAI RECORD: 0x04 0x01

Il primo valore letto è 04.

Essendo nella condizione iniziale lo si usa sia come indice della matrice che del vettore.

La coppia Matrice/Vettore che si utilizza è nella

posizione 1 dell'elenco sopra riportato.

L'indice di riga della matrice A viene determinato dall'elemento letto, vale a dire Riga A =04, cioè la quarta riga.

L'indice di colonna della matrice A viene determinato dal valore contenuto dal vettore B nella posizione corrispondente all'elemento letto, vale a dire Colonna A =B[04]=X. Lo stream non viene pertanto riconosciuto.

3) ESEMPIO DI RICONOSCIMENTO NEL CASO IN CUI LO STREAM SIA DIVERSO DAI RECORD: 0x01 0x05 0x03

Il primo valore letto è 01.

Essendo nella condizione iniziale lo si usa sia come indice della matrice che del vettore.

La coppia Matrice/Vettore che si utilizza è nella posizione 1 dell'elenco sopra riportato.

L'indice di riga della matrice A viene determinato dall'elemento letto, vale a dire Riga A =01, cioè la prima riga.

L'indice di colonna della matrice A viene determinato dal valore contenuto dal vettore B nella posizione corrispondente all'elemento letto, vale a dire Colonna A =B[0x01]=0, cioè la zeresima colonna.

Si andrà pertanto a leggere il valore riportato in A[1, 0] e cioè 0. Tale valore costituisce il

prossimo indice del vettore B.

Si andrà poi a leggere l'azione riportata in A[1, 0] vale a dire il valore numerico corrispondente all'azione next(1).

Si andrà pertanto al prossimo valore e si utilizzerà quindi la coppia Matrice/Vettore che è nella posizione 2 dell'elenco sopra riportato.

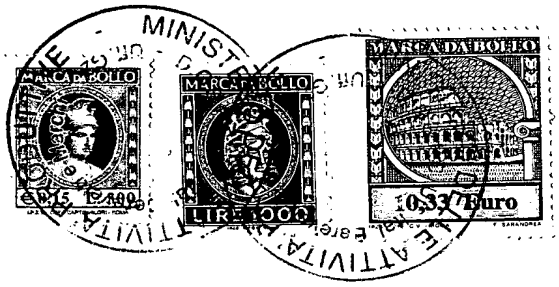
Il valore letto è 05.

L'indice di riga della matrice A viene determinato dall'elemento letto, vale a dire Riga A =05, cioè la quinta riga.

L'indice di colonna della matrice A viene determinato dal valore contenuto dal vettore B nella posizione corrispondente al valore riportato in A[1, 0] ottenuto nel passo precedente. Colonna A =B[0]=0.

Bisognerà pertanto andare a leggere il valore riportato in A[5, 0] e cioè X. Lo stream non viene pertanto riconosciuto.

Il riconoscitore di pattern, quindi, usando una tecnica ad accesso diretto su matrici, facilmente implementabile in un microchip è in grado di riconoscere in modo completamente deterministico l'accettabilità o la non accettabilità dello stream in ingresso in un numero di accessi a matrici e vettori, pari al numero degli elementi riconosciuti nello



stream stesso

* * *

Si farà d'ora in poi nuovamente riferimento alla figura 9.

Elemento 205:

E' la componente di rilevamento ed acquisizione delle trame di comunicazione. Tramite questo apparecchio, un esempio del quale è già stato descritto in dettaglio con riferimento alle precedenti figure da 4 ad 8B, viene resa possibile l'acquisizione di dati anche a livello applicativo, vale a dire l'informazione relativa ai livelli 4-7 dello standard OSI. Tale apparecchio potrà accettare comandi quali i comandi CONNECT, SEND, RECEIVE e CLOSE nel caso in cui si debbano gestire e coordinare protocolli applicativi di alto livello.

Elemento 206 (Controllo accessi):

Questo elemento, a partire dal risultato di riconoscimento operato dall'elemento 204, esegue l'azione di inoltro associata a tale riconoscimento oppure l'azione di rifiuto associata al non riconoscimento.

Nel caso di accettazione la trama di comunicazione verrà inoltrata al server di riferimento.

Nel caso di rifiuto, la trama di comunicazione verrà rispedita al mittente, unitamente alle eventuali motivazioni del rifiuto. Infatti, grazie alla struttura <oggetto>/<azione> adottata, sarà possibile associare azioni, anche complesse, quali la costruzione di stream di risposta.

Elemento 207 (Coordinamento accessi):

Questo elemento, a partire dal risultato di riconoscimento operato dall'elemento 204, esegue l'azione di coordinamento associata a tale riconoscimento.

Tale azione di coordinamento riguarda l'individuazione dei parametri da inviare al server per il coordinamento richiesto, l'individuazione del server, la formattazione dei parametri da inviare, l'invio dei parametri, l'acquisizione della risposta dal server e l'inoltro della risposta ottenuta all'elemento 204 per l'eventuale prosecuzione del riconoscimento.

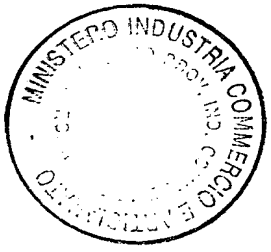
Questo approccio è reso possibile dall'uso della seconda notazione introdotta in quanto grazie a questa notazione è possibile associare azioni, anche complesse, quali la costruzione di stream da inoltrare a particolari server accessibili da rete. L'elemento di coordinamento ha un suo senso compiuto quando si

utilizzi lo strumento per la gestione di comunicazioni tra applicazioni e quindi su protocolli ad alto livello (quali quelli tra applicazioni client e server trasportati a livello TCP). In tal caso, infatti, lo strumento, grazie alle azioni associate al riconoscimento degli stream di ingresso, può operare trasformazioni dello stream per il suo reinoltro ad altre applicazioni server che prevedono protocolli applicativi diversi. Un tipico caso è quando si deve gestire l'interoperabilità e la cooperazione applicativa in un contesto eterogeneo e devono coesistere differenti "server applicativi" o differenti dispositivi di mediazione (broker) (facendo in questo caso riferimento anche alle diverse implementazioni di CORBA -Common Object Request Broker Architecture- che non sono mai completamente compatibili tra loro) a fronte di applicazioni utente (client) spesso progettate per dialogare utilizzando protocolli applicativi non aggiornati.

La presente invenzione è stata fin qui descritta con riferimento ad una sua forma di realizzazione illustrata a scopo esemplificativo e non limitativo.

E' da intendersi inoltre che altre siano le forme di realizzazione possibili rientranti nell'ambito della presente privativa industriale.

Giorgio Strini
(Iscr. Albo n. 452 BM)



RM 98A000542

RIVENDICAZIONI

1. Dispositivo di controllo di accessi in rete tramite il riconoscimento deterministico di trame applicative che soddisfano un insieme di regole predefinite comprendente:

- mezzi (205) di rilevamento ed interpretazione delle trame applicative da riconoscere;
- mezzi (201) di memorizzazione di regole predefinite;
- mezzi (202) di compilazione delle regole predefinite in una struttura dati ad accesso diretto;
- mezzi (203) di memorizzazione di detta struttura dati ad accesso diretto; e
- mezzi (204) di confronto tra le trame applicative da riconoscere e detta struttura dati ad accesso diretto,

in cui il riconoscimento può essere effettuato su qualsiasi componente della trama ed in cui la struttura dati ad accesso diretto permette di ottenere un tempo di accesso sostanzialmente indipendente dalla numerosità delle regole.

2. Dispositivo di controllo di accessi secondo la rivendicazione 1, caratterizzato dal fatto che detti mezzi (202) di compilazione delle regole predefinite comprendono:

- mezzi di conversione delle regole predefinite in un insieme di sequenze elementari di identificatori

numerici; e

- mezzi di compressione dell'insieme di sequenze così ottenute in una struttura dati ad accesso diretto.

3. Dispositivo di controllo di accessi secondo la rivendicazione 1 o 2, caratterizzato dal fatto di comprendere inoltre mezzi di inoltro della trama applicativa quando riconosciuta e mezzi di restituzione al mittente della trama applicativa quando non riconosciuta.

4. Dispositivo di controllo di accessi secondo la rivendicazione 3, caratterizzato dal fatto che detti mezzi di restituzione al mittente della trama applicativa quando non riconosciuta prevedono la restituzione di informazioni relative ai motivi del mancato inoltro.

5. Dispositivo di controllo di accessi secondo una qualsiasi delle rivendicazioni precedenti, caratterizzato dal fatto che le regole predefinite sono memorizzabili quale coppia di campi <oggetto>/<azione>.

6. Dispositivo di controllo di accessi secondo la rivendicazione 5, caratterizzato dal fatto che le regole predefinite sono memorizzate quale coppia di campi <tipo di dato>/<valore dato>.

7. Dispositivo di controllo di accessi secondo la

rivendicazione 5 o la rivendicazione 6, caratterizzato dal fatto che le regole predefinite contengono uno o più valori jolly.

8. Dispositivo di controllo di accessi secondo la rivendicazione 5, caratterizzato dal fatto che il campo <azione> fa riferimento all'insieme minimo di comandi

- **Push**

<valore>

<variabile>

<posizione di lettura>

<valore alla posizione di lettura>

- **Pop**

<variabile>

<posizione di lettura>

<nella posizione di lettura>

- **And**
- **Mul**
- **Add**
- **Equal**
- **Next**
- **F_send_all**
- **F_dynamic.**

9. Dispositivo di controllo di accessi secondo le rivendicazioni 2 e 5, caratterizzato dal fatto che la

struttura dati ad accesso diretto è rappresentata tramite una struttura matriciale comprendente campi oggetto e campi azione.

10. Dispositivo di controllo di accessi secondo una qualsiasi delle rivendicazioni precedenti, caratterizzato dal fatto che i mezzi (205) di rilevamento ed interpretazione delle trame applicative comprendono:

- un dispositivo (9) di rilevamento di pacchetti di dati ad un livello corrispondente al livello OSI 2 comprendenti frame di controllo e frame di informazione, in cui i frame di controllo ed informazione comprendono una parte di intestazione ed una parte di corpo, detta parte di intestazione essendo atta a permettere la distinzione tra un frame di informazione ed un frame di controllo;
- una unità di controllo (15) ricevente in ingresso i dati provenienti dal dispositivo di rilevamento (9) e comprendente mezzi atti a discriminare i frame di controllo dai frame di informazione;
- una unità (16) di datazione collegata all'unità di controllo (15) e tale da associare un istante temporale di rilevamento ai frame di controllo ed ai frame di informazione;
- una unità (17) di memorizzazione di dati

discriminati atta a memorizzare i frame di controllo, i frame di informazione e l'istante temporale di rilevamento degli stessi, collegata in maniera bidirezionale all'unità di controllo (15); e

- una unità (18) di memorizzazione di dati predeterminati, collegata in maniera bidirezionale all'unità di controllo (15), detti dati predeterminati rappresentando possibili interpretazioni dei frame di informazione o di controllo contenuti nell'unità (17) di memorizzazione di dati discriminati ed essendo atti ad essere confrontati, tramite l'unità di controllo (15), con i dati contenuti nella parte di corpo dei frame di informazione o di controllo memorizzati nell'unità (17) di memorizzazione di dati discriminati, in maniera tale da permettere:

- un ordinamento temporale e secondo il tipo di comunicazione delle parti di corpo dei frame di controllo e di informazione; e

- una ricostruzione di alberi applicativi arricchiti di informazioni di tipo statistico secondo il tipo di comunicazione, in maniera da permettere certificazione delle comunicazioni ed il rilevamento di eventuali anomalie.

11. Dispositivo di controllo di accessi secondo la rivendicazione 10, caratterizzato dal fatto che il

dispositivo (9) di rilevamento dati comprende:

- un ricevitore dei dati di sorgente (12);
- un ricevitore dei dati di destinazione (13); e
- una interfaccia di connessione (14) atta a ricevere i segnali provenienti dal ricevitore dei dati di sorgente (12) e dal ricevitore dei dati di destinazione (13) ed a trasmettere gli stessi verso l'unità di controllo (15).

12. Dispositivo di controllo di accessi secondo la rivendicazione 10 o 11, caratterizzato dal fatto che la ricostruzione di detto albero applicativo arricchito di informazioni di tipo statistico avviene tramite confronto reciproco di ciascuna parte di corpo dei frame di informazione con le altre.

13. Dispositivo di controllo di accessi secondo una qualsiasi delle rivendicazioni da 10 a 12, caratterizzato dal fatto che la ricostruzione di detto albero applicativo arricchito di informazioni di tipo statistico avviene tramite confronto di ciascuna sequenza di parti di corpo dei frame di informazione con un insieme di sequenze predeterminate, rappresentanti possibili interpretazioni di sequenze di frame di informazione o di controllo contenuti nell'unità (17) di memorizzazione di dati discriminati, dette sequenze predeterminate essendo

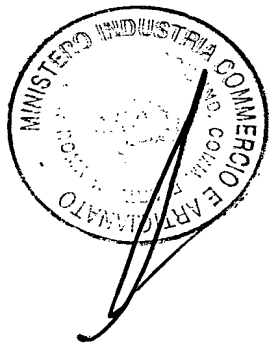
contenute in detta unità (18) di memorizzazione di dati predeterminati.

14. Dispositivo di controllo di accessi secondo una qualsiasi delle rivendicazioni da 10 a 13, caratterizzato dal fatto che detta unità (16) di datazione è di tipo a tempo assoluto, in particolare via radio o satellitare.

15. Dispositivo di controllo di accessi sostanzialmente come descritto in precedenza con riferimento ai disegni annessi.

p.p. ALASI di Arcieri Franco & C. s.a.s.

Giorgio Strini
(Iscr. Albo n. 452 BM)



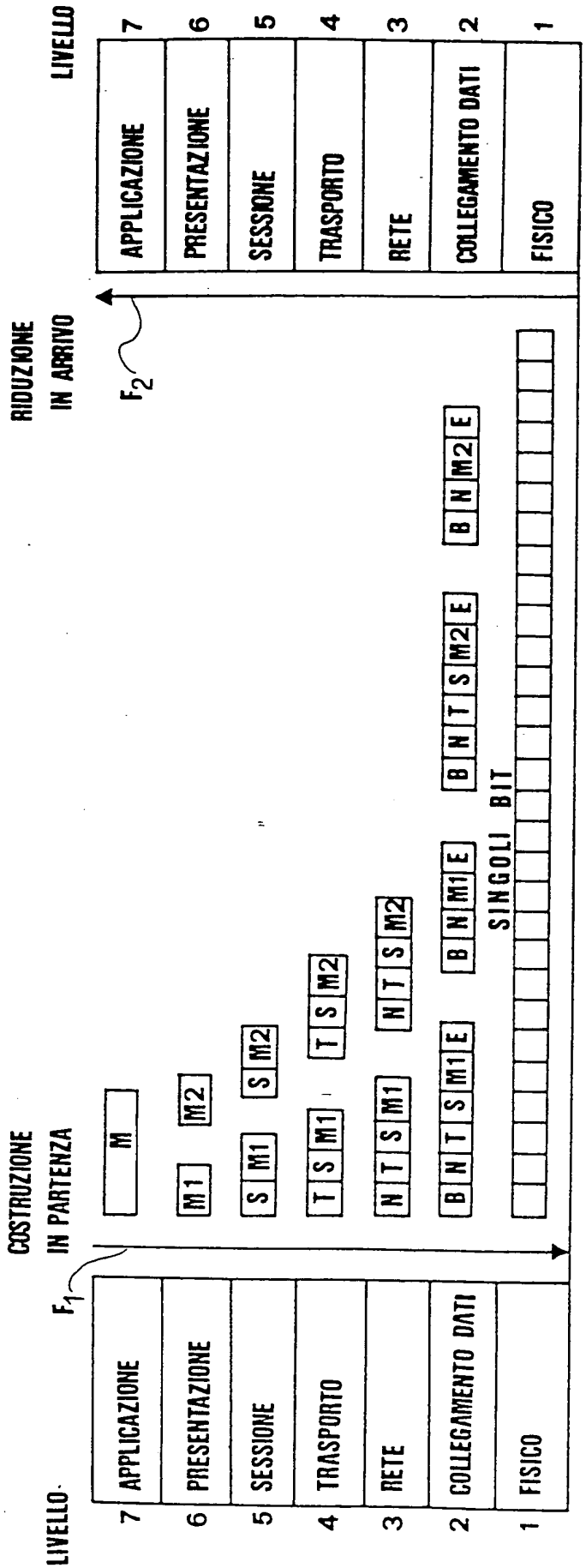


FIG.1

RMR1099

p.p. ALASI di Arcieri Franco & C. s.a.s.

Gilberto Tonon
(Iscr. Albo n. 83 BM)

R M R 1099

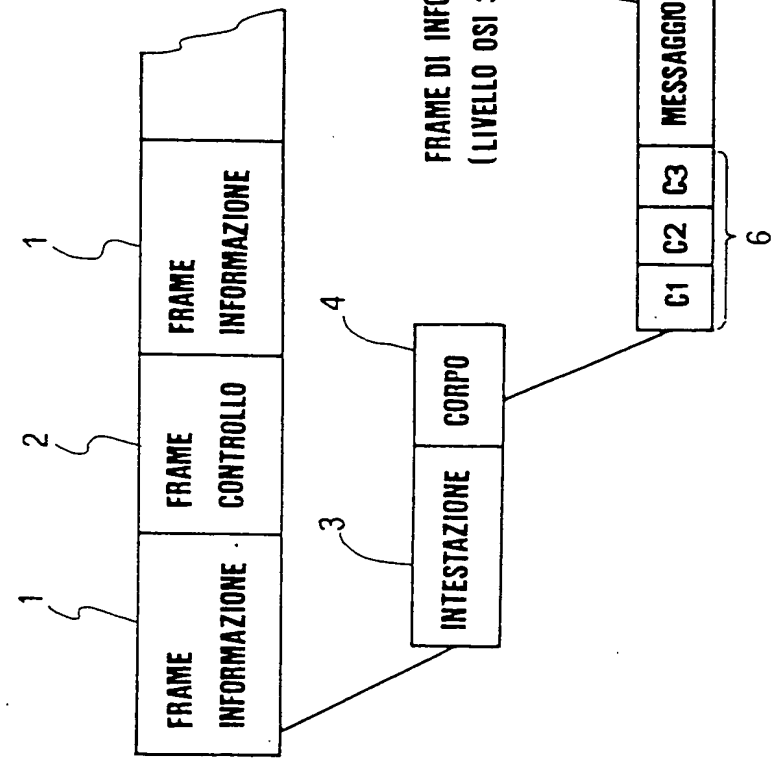


FIG.2

p.p. ALASI di Arcieri Franco & C. s.a.s.

Gilberto Tonon
(Iscr. Albo n. 83 BM)

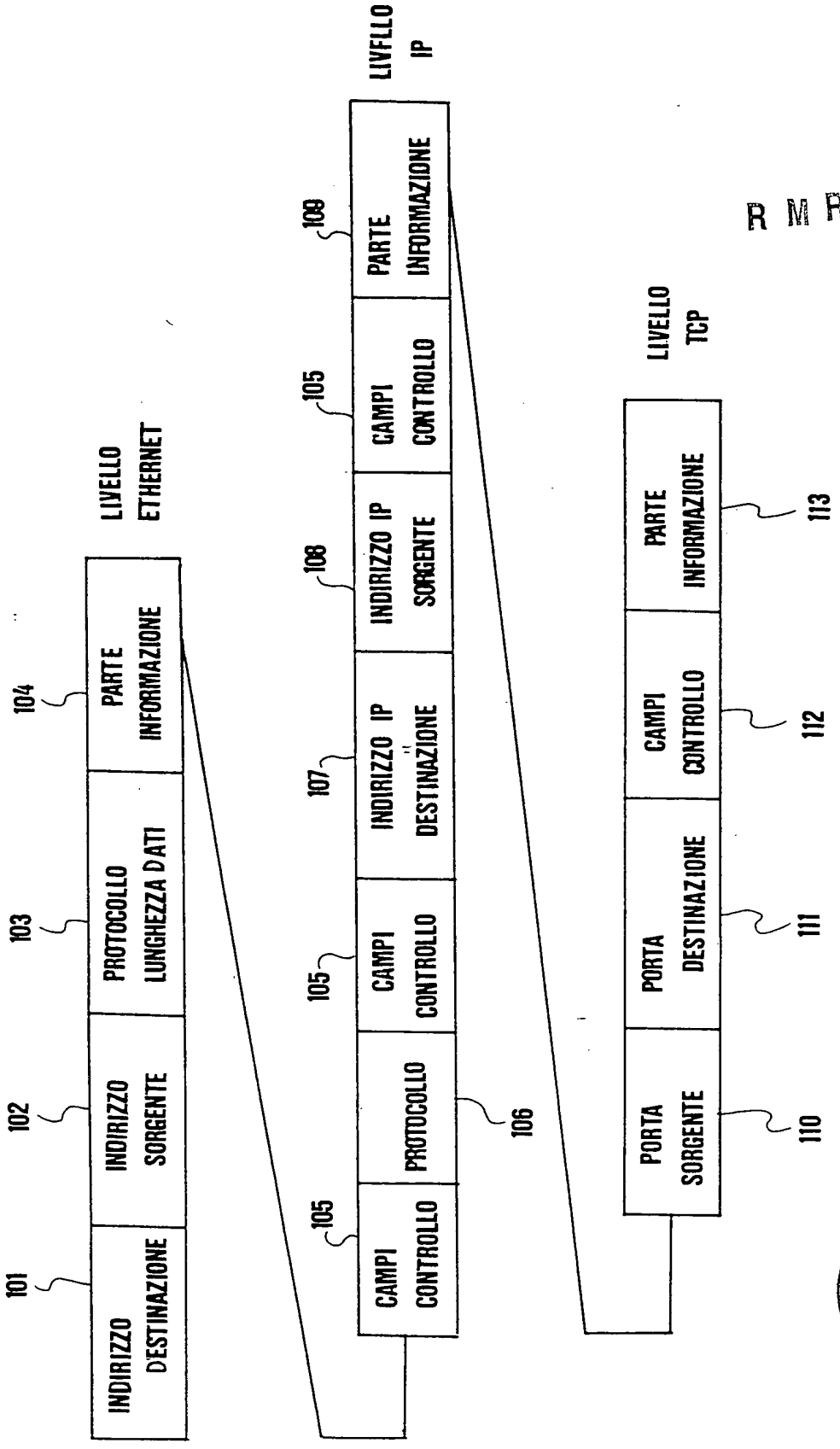


FIG.3

RMR1099

p.p. ALASI di Arcieri Franco & C. s.a.s.

Oliviero Tonon
(Iscr. Albo n. 83 BM)

R M R 1 0 9

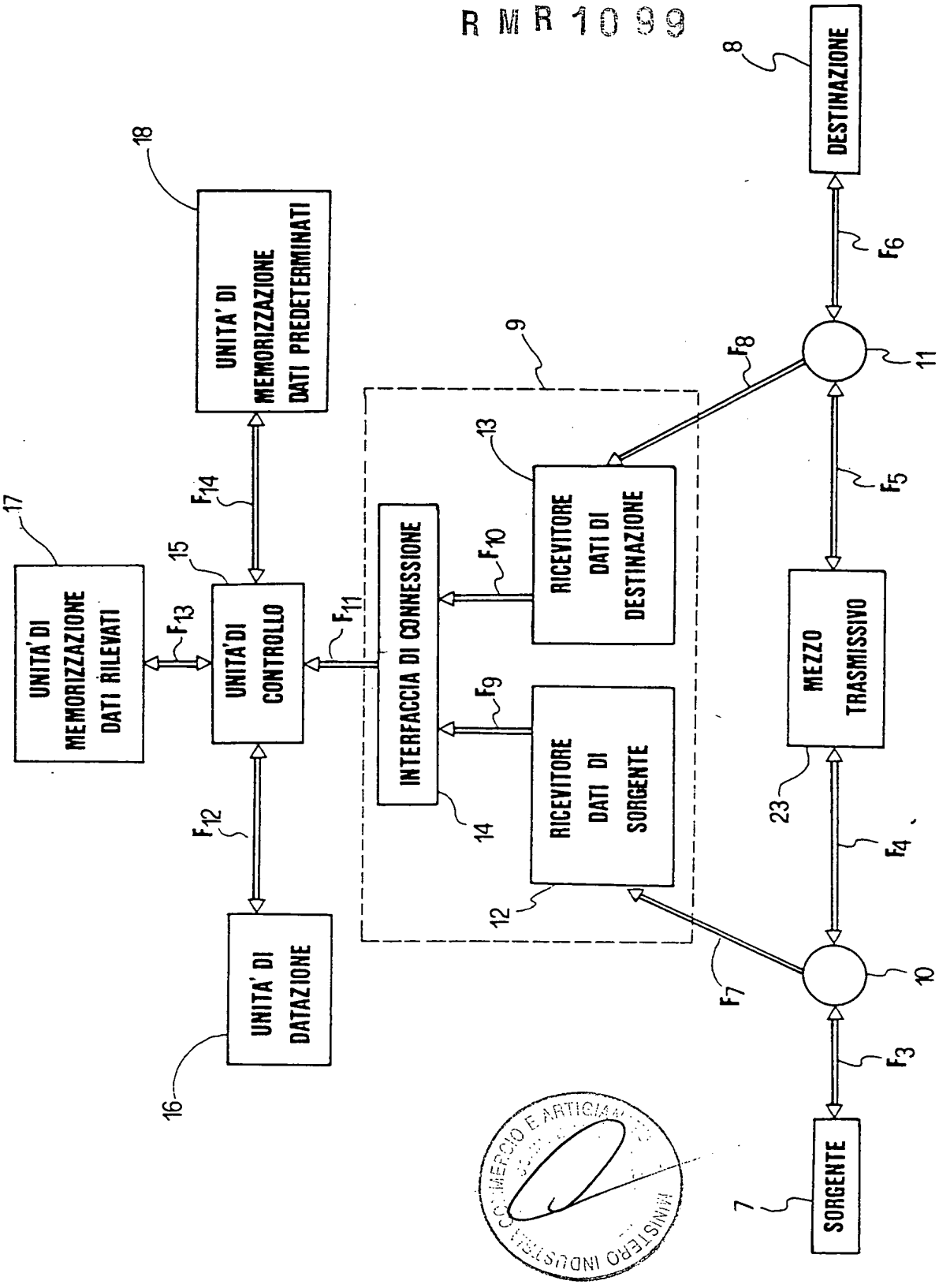


FIG.4

Gilberto Tonon
Iscr. Albo n. 83 BM

p.p. ALASI di Arcieri Franco & C. s.a.s.

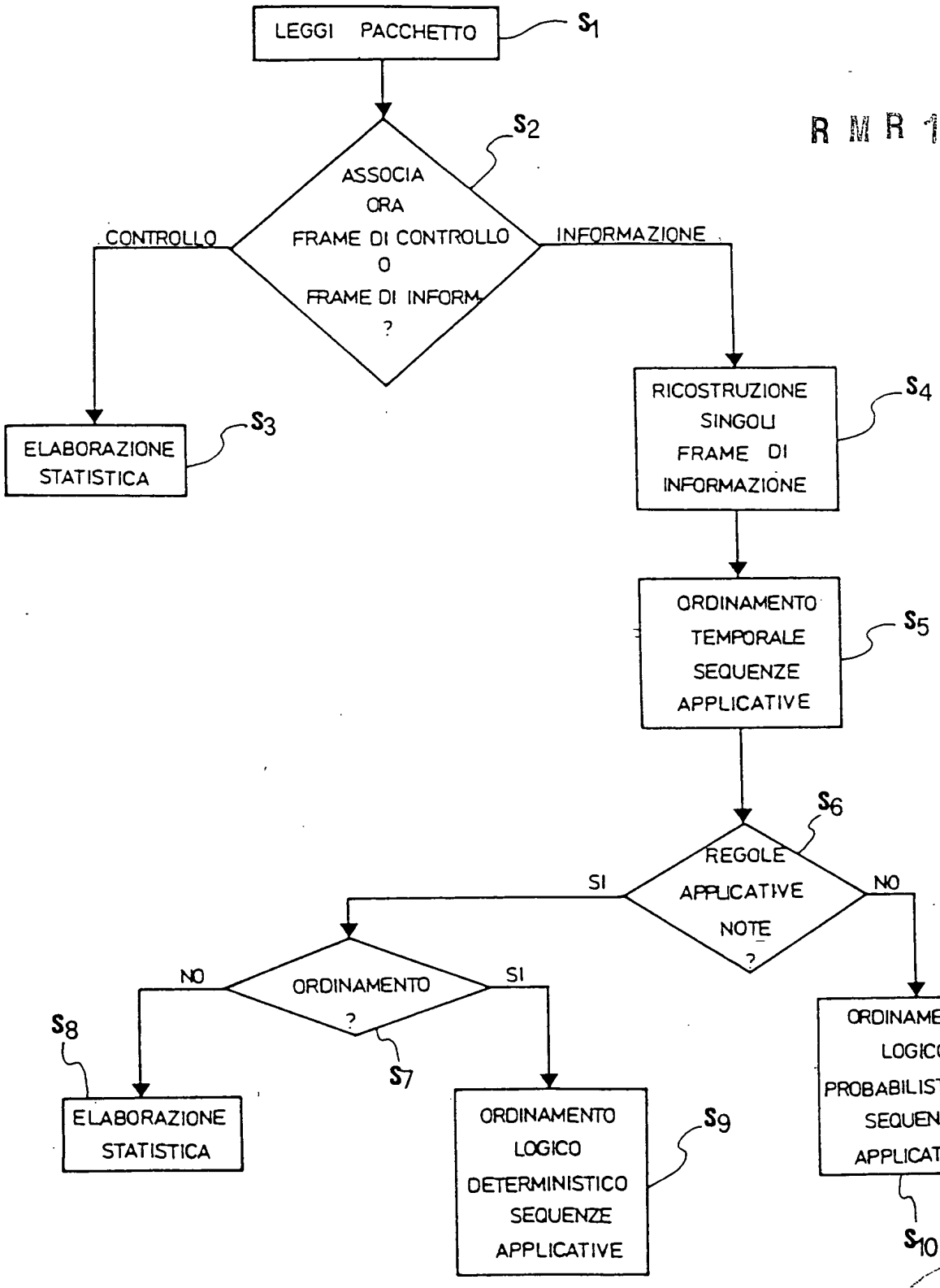
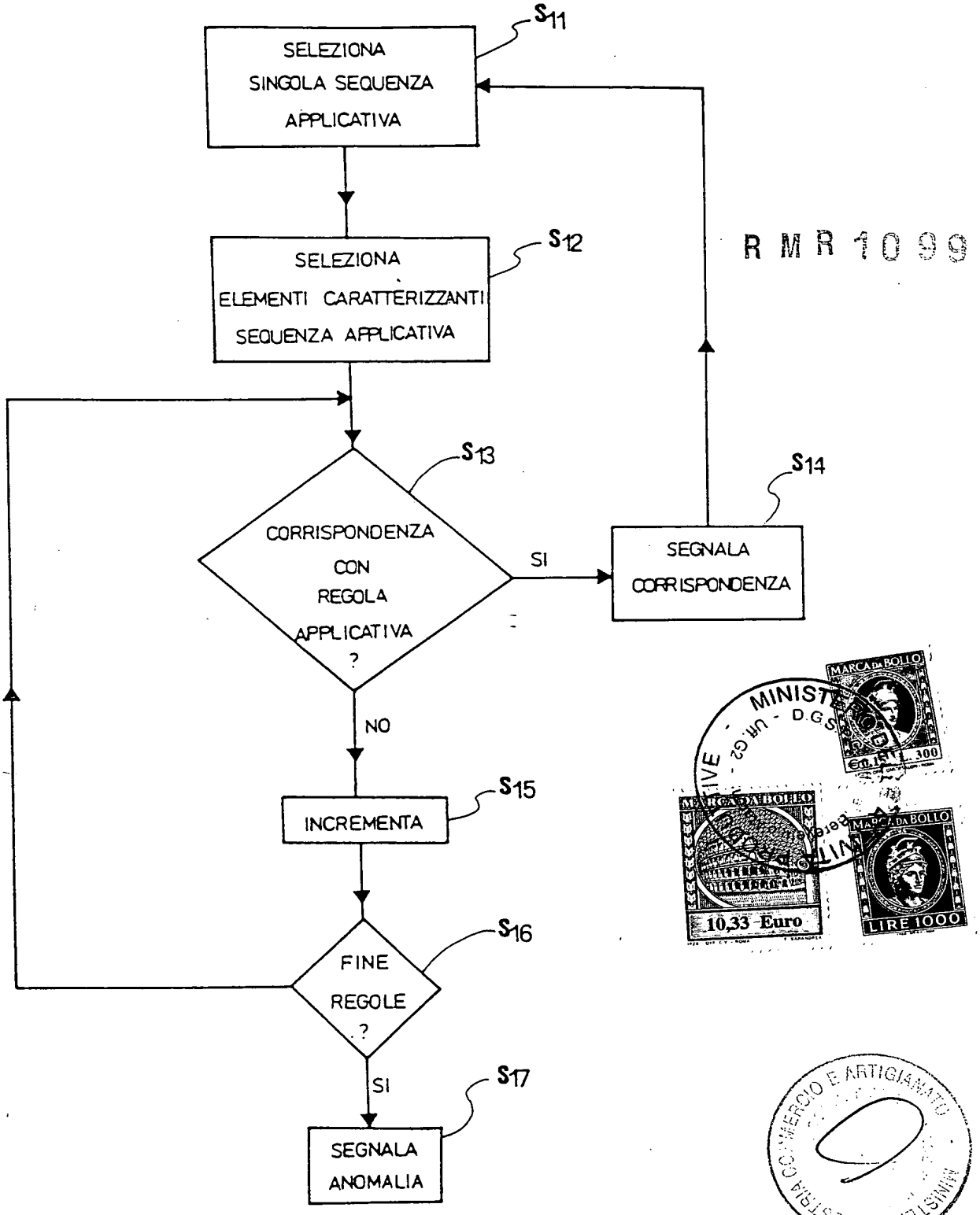


FIG.5

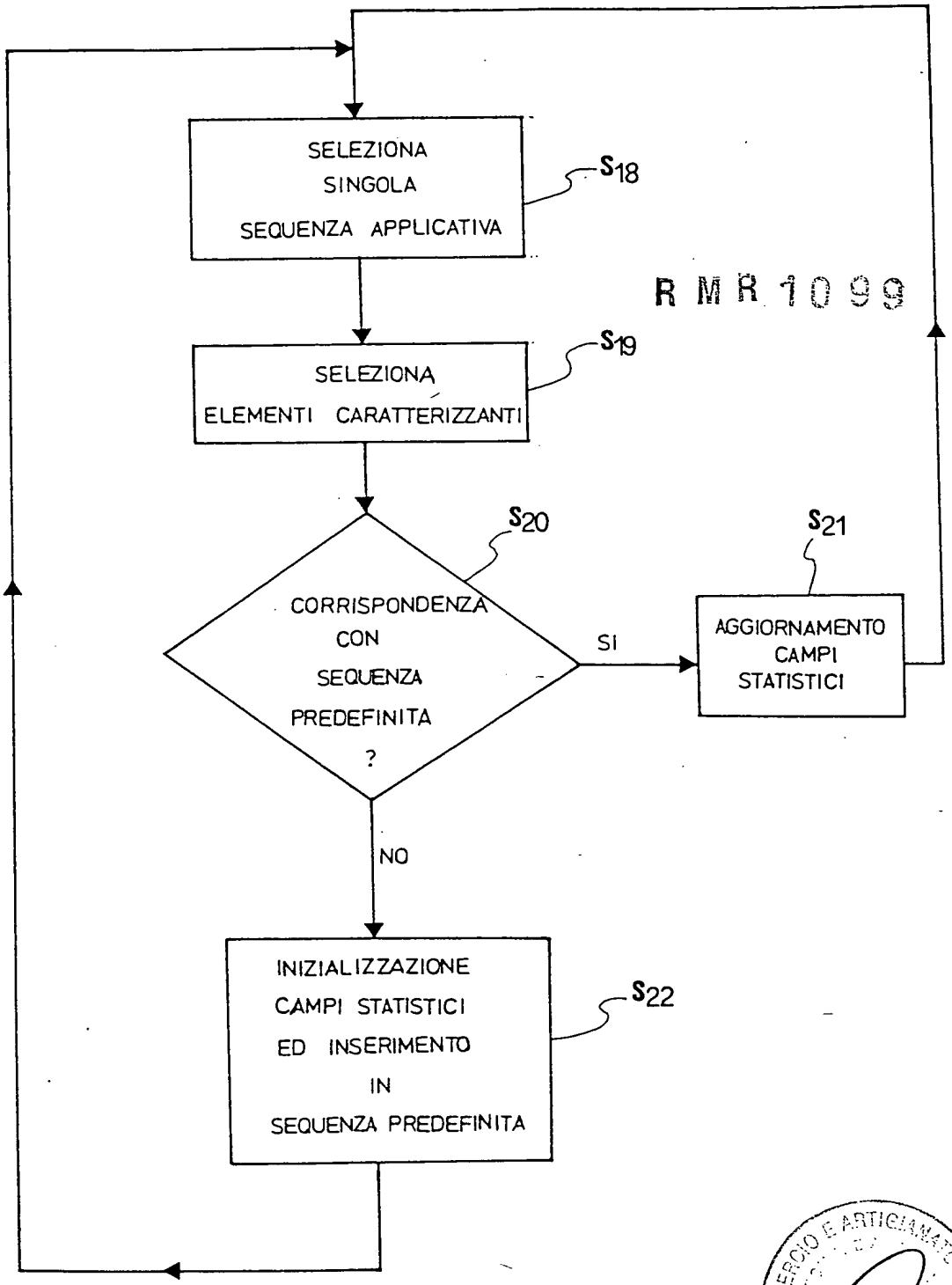
p.p. ALASI di Arcieri Franco & C. s.a.s.

Giiberto Tonon
(Iscr. Albo n. 83 BM)



RMR1099

FIG.6



RMR1099

FIG.7

p.p. ALASI di Arcieri Franco & C. s.a.s.

Gilberto Tonon
Iscr. Albo n. 83 BM

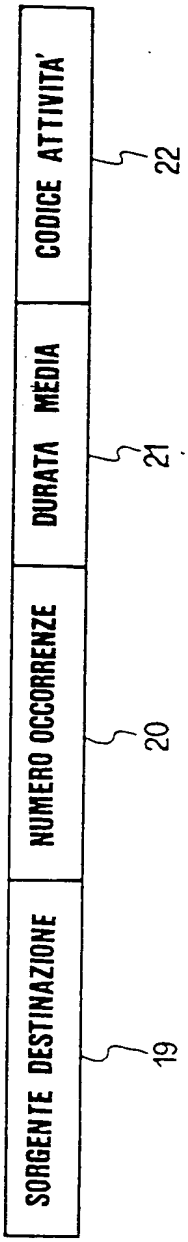


FIG.8A

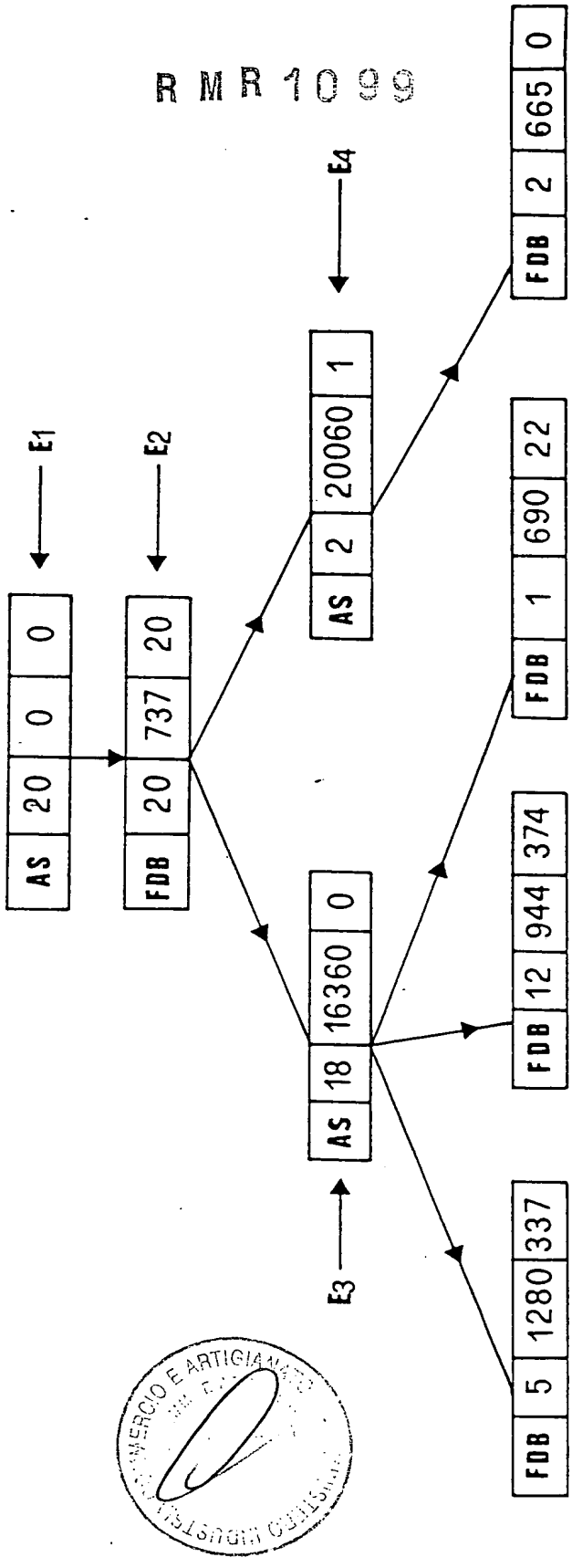


FIG.8B

R M R 1099

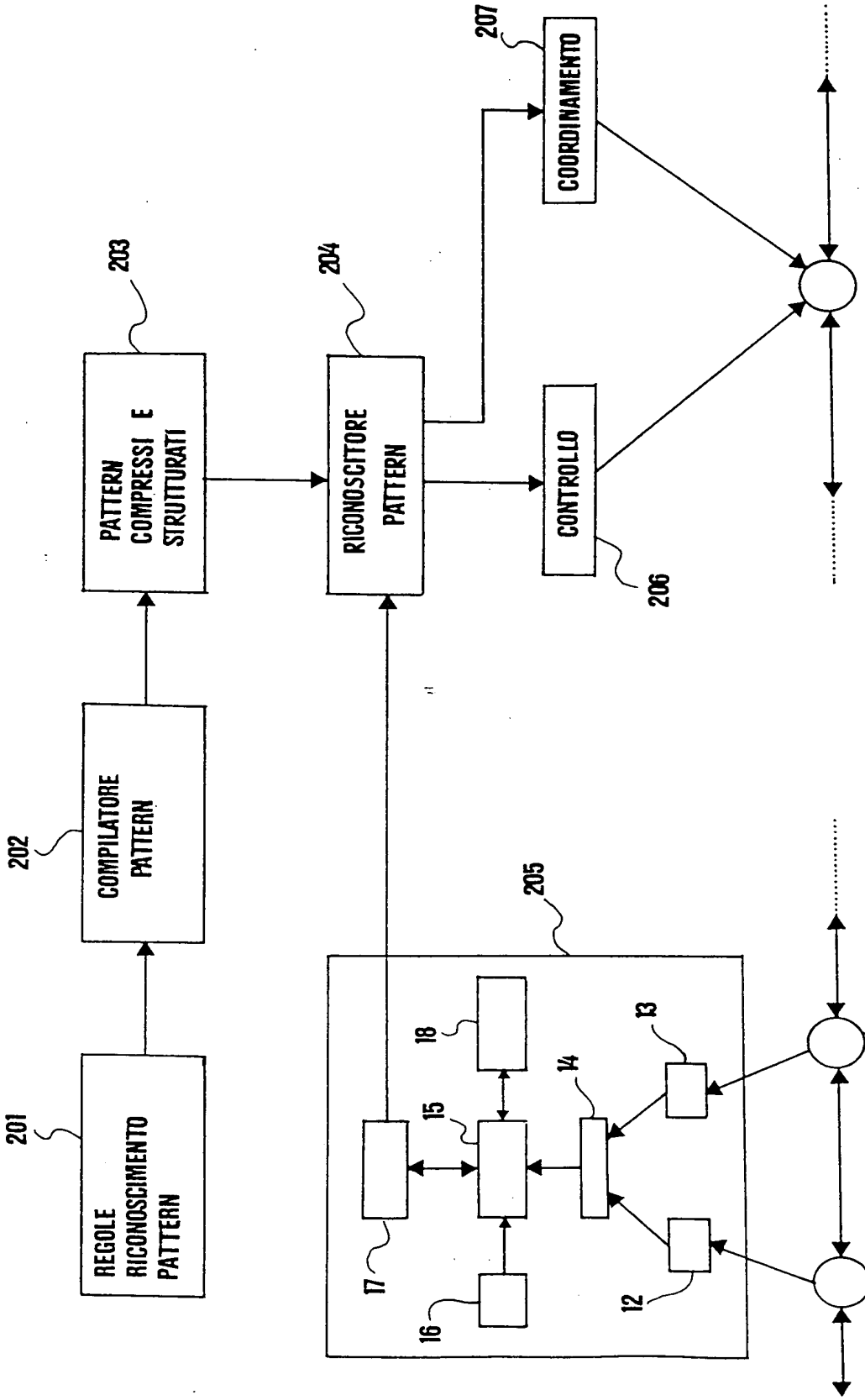


FIG.9

p.p. ALASI di Arcieri Franco & C. s.a.s.

Gilberto Tonon
(Iscr. Albo n. 83 BM)

RMR 1099

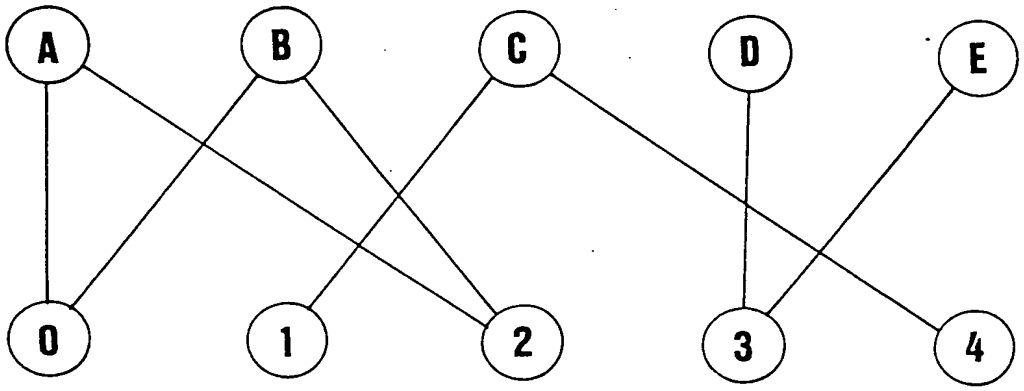


FIG.10A

	0	1	2	3	4
A	×		×		
B	×		×		
C		×			×
D			×		
E			×		

FIG.10B

p.p. ALASI di Arcieri Franco & C. s.a.s.

Roberto Tonon
(Iscr. Albo n. 83 BM)

RMR 1059

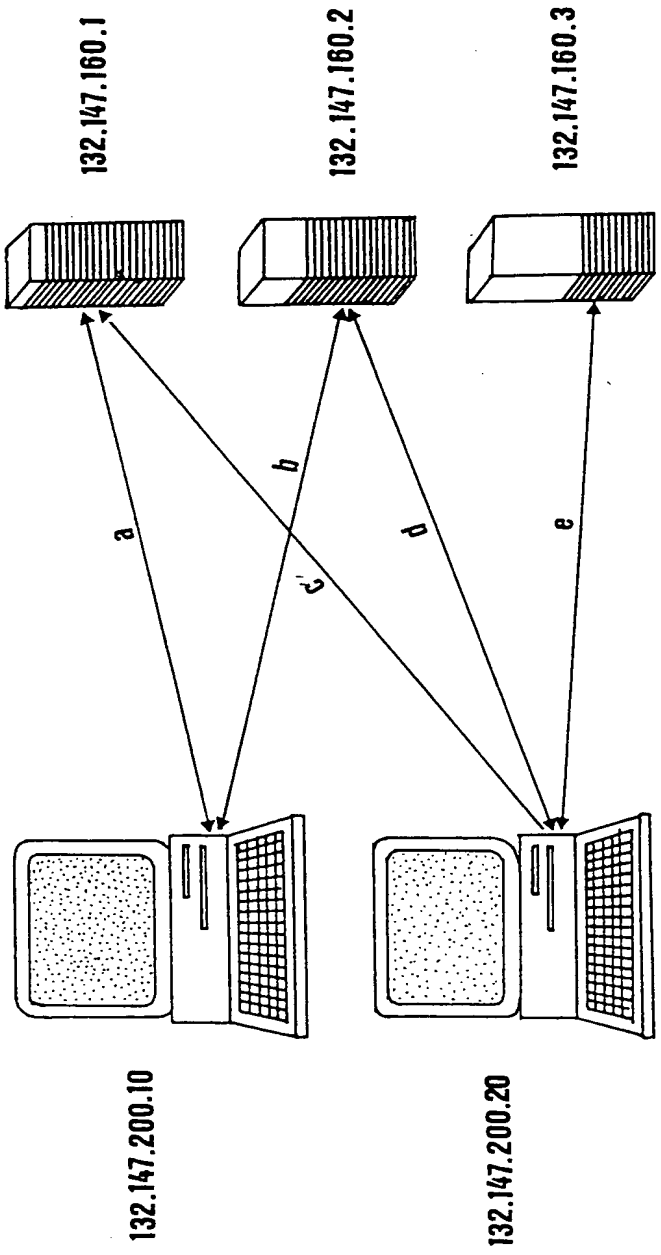


FIG.11

p.p. ALASI di Arcieri Franco & C. s.a.s.

Stefano Fozon
(Iscr. Albo n. 324 BM)

RMR1099

	0	1	2	3	4	5	6	7	8	9	0A	0B	0C
1:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(0A,400F)	(84,4010)	(93,4011)	(A0,4012)	(01,4013)	(00,8002)	(50,8003)
2:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(0A,400F)	(84,4010)	(93,4011)	(A0,4012)	(02,4013)	(00,8002)	(19,8003)
3:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(0A,400F)	(84,4010)	(93,4011)	(A0,4012)	(02,4013)	(00,8002)	(89,8003)
4:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(0A,400F)	(84,4010)	(93,4011)	(A0,4012)	(02,4013)	(00,8002)	(8A,8003)
5:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(0A,400F)	(84,4010)	(93,4011)	(A0,4012)	(02,4013)	(00,8002)	(8B,8003)
6:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(14,400F)	(84,4010)	(93,4011)	(A0,4012)	(01,4013)	(00,8002)	(14,8003)
7:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(14,400F)	(84,4010)	(93,4011)	(A0,4012)	(01,4013)	(00,8002)	(15,8003)
8:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(14,400F)	(84,4010)	(93,4011)	(A0,4012)	(01,4013)	(00,8002)	(17,8003)
9:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(14,400F)	(84,4010)	(93,4011)	(A0,4012)	(02,4013)	(00,8002)	(19,8003)
0A:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(14,400F)	(84,4010)	(93,4011)	(A0,4012)	(02,4013)	(00,8002)	(50,8003)
0B:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(14,400F)	(84,4010)	(93,4011)	(A0,4012)	(03,4013)	(00,8002)	(50,8003)
0C:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(14,400F)	(84,4010)	(93,4011)	(A0,4012)	(03,4013)	(00,8002)	(A1,8003)
0D:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(14,400F)	(84,4010)	(93,4011)	(A0,4012)	(03,4013)	(00,8002)	(A2,8003)
0E:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(14,400F)	(84,4010)	(93,4011)	(A0,4012)	(03,4013)	(08,8002)	(01,8003)
0F:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(14,400F)	(84,4010)	(93,4011)	(A0,4012)	(03,4013)	(00,8002)	(17,8003)
10:	(08,000C)	(06,000D)											
11:	(08,000C)	(00,000D)	(01,4009)										

FIG.12

p.p. ALASI di Arcieri Franco & C. s.a.s.

Claudio Turon
(iscr. Albo n. 83 BM)

"NETWORK ACCESS CONTROL DEVICE THROUGH FAST RECOGNITION OF APPLICATION FRAMES SATISFYING A SET OF PREDETERMINED RULES"

DESCRIPTION

The present invention provides a network access control device through fast recognition of application frames satisfying a set of predetermined rules.

In particular, the device according to the invention allows both monitoring and interpretation of application protocols for network data transmission systems and the comparison with a set of control patterns of every monitored and interpreted communication frame. If a frame is recognized, the device allows access to the service. If the frame is not recognized, the device denies access to the service.

In the present description, for "pattern" (or access rule) the recognition statement of a particular communication frame will be intended.

Preferably, such statement will be intended as a set of <data type>/<data value> pairs assumed by the fields inside the communication frame. The <data type>/<data value> pairs are specified according to the various communication layers inside the communication frame concerning both the control and the information portion. In the present description, by way of example, communication frames of the HTTP (Internet browsing services) kind will be illustrated.

Network access control devices are known and can be divided into two large categories:

1) In a first category, the various access rules are represented by means of multidimensional matrices represented in a non-compressed form, using simple languages to access said matrices. The disadvantage of such a representation is given by the high memory occupation: a 10-dimensional matrix with 100 elements per dimension has a memory occupation of 100^{10}.

2) In a second category, the various access rules are represented by means of multidimensional matrices represented in a compressed form. The access to said matrices is not of a direct type. Such a manner has the disadvantage of requiring the use of high level languages, which determine the particular procedure to be activated in response to the recognition of an access rule by means of test and comparison operators. The particular control structures thus used burden the interpretation process, making it inefficient. However, the realization of generalized methods for information structure recognition on fast technologies (firmware) proves to be difficult, if not altogether impossible.

Furthermore, in both types of devices of the known art there is the disadvantage that the recognition of the communication frames cannot be based on any frame component, but exclusively on frames at a non-application layer.

The present invention overcomes such disadvantages of the prior art, as it provides a network access control device through deterministic recognition of application frames satisfying a set of predetermined rules comprising:

- means for monitoring and interpretation of the application frames to recognize;
- means for storing predetermined rules;
- means for compiling the predetermined rules in a direct access data structure;
- means for storing said direct access data structure; and
- means for comparing the application frames to be recognized with said direct access data structure,

wherein the recognition is able to be performed on any frame component and the direct access data structure allows an access time substantially independent from the number of rules.

As it is known, a direct network access data structure allows the access to the i-th element without necessarily having to access to the preceding elements, as it occurs instead with sequential access data structures. Known examples of direct access data structures are vectors, matrices, correspondence tables, a memory of a processor etc.

A first advantage of the access control device according to the present invention is given by the flexibility wherewith a recognition pattern can be realized. In fact, by virtue of the use of the apparatus for monitoring and interpretation of application protocols, described in detail herebelow, the recognition of the communication frames can be based on any of the components of the frame, both on the control portion and on the information portion. Therefore, a recognition pattern can be realized (and therefore the access can be restricted) based on the contents of the exchanged information between client and server and not only based on the used network addresses and services.

A second advantage of the device according to the present invention is given by its capability to manage a very high number of patterns (in the order of millions) without any decay in the performances.

In fact, in a context where a high number of users, servers and application services on the same servers is involved, when it is desirable to directly

manage (or to control and document) the accessibility of each user to the single server and to the application service thereby provided, the pattern number grows quadratically. For instance, given 1000 users on the territory and 100 servers of which it is desirable to manage and control the accesses, 1000*100=100.000 patterns are generated. This number further increases when it is desirable to manage and/or control the access to the application for each single server, the pattern number growing to the order of the millions in the actual cases on of middle/big-dimensioned structures.

Such a pattern number is by all means acceptable for the device of the present invention.

In fact, the recognition of the acquired communication frames is based on a deterministic access algorithm (hence neither heuristic nor probabilistic) ensuring an access time that is constant and independent (under any input) from the pattern number.

Then the access control device performs for each correctly recognized frame the coordinating operation associated to recognition. Once the recognition has occurred, in fact, the device will activate a TCP/IP layer (or layers corresponding to other protocols) communication with the server application individuated as result of the recognition, even a partial one, of the information component of the access frame, providing as parameters part of the already recognized or yet unprocessed information component. The sending modes (parameter sending format, number of parameters to send, application to activate, etc.) are associated to the recognition action and are therefore stored in the patterns.

The access control device according to the present invention can be configured to operate both in positive logic and in negative logic.

In positive logic, all the frames that meet the recognition patterns will be considered as accepted and therefore brought to destination or are subjected to a coordinating, control and/or documentation action.

In negative logic, all the frames that do not meet the recognition patterns will be considered as accepted and therefore brought to destination. All the recognized frames will not be brought to destination.

The present invention will be illustrated herebelow by referring to a preferred embodiment thereof, explained by way of a non-limiting example. Reference to figures of the annexed drawings will be made, wherein:

figure 1 shows a schematic diagram of the OSI standard;

figure 2 shows a schematic view of the type of data used on communication networks;

figure 3 shows a schematic view of the type of data used on communication networks with reference to the TCP/IP protocol ;

figure 4 shows a block diagram of the apparatus for monitoring and interpretation belonging to the access control device according to the present invention;

figure 5 shows a flow chart explaining the operation of the component in figure 4;

figures 6 and 7 show additional flow charts for the understanding of what disclosed with reference to figure 5;

figures 8A and 8B show an example of application tree containing statistic information obtained by means of the component in figure 4;

figure 9 shows a block diagram of the access control device according to the present invention;

figures 10A and 10B show examples of the logical correspondence between bipartite graph and bidimensional matrix;

figure 11 contains an example of specification of predetermined rules; and

figure 12 shows a matricial representation of sequences of numerical identificators.

Data transmission from a source device to a destination device can occur in different manners. However, to ensure a data exchange having the lowest possible chance of errors it is necessary to adopt a series of rules or control procedures. Said rules or procedures are known as "communication protocols".

A well known communication protocol is the "Open System Interconnection" (OSI) of the International Standards Organization (ISO). Said protocol is divided into seven layers, shown in figure 1. Layer 7 (application) on the source side contains information related to the sole message (M) to be sent to the destination side. The successive layers on the source side add control information to the message: layer 6 (presentation) divides the data of the original message into blocks (M1 and M2); layer 5 (session) adds a title (S) to indicate the sender, the receiver and some information related to the sequence; layer 4 (transport) adds information (T) related to the logic connection between the sender and the receiver; layer 3 (network) adds information related to the path (N) and divides the message into packets representing the standard communication unit in a network; layer 2 (data

link) adds a title portion (B) and a tail portion (E) to the message to ensure the correct order of the various packets and to correct transmission errors; the single message bits and control information bits added by the various layers are transmitted on the physical medium through layer 1. The downward pointing arrow F1 on the sender side indicates the manner according to which the outgoing message is constructed. Every addition to the message is verified and removed from the corresponding layer on the destination side. The upward pointing arrow F2 on the destination side indicates the manner according to which the incoming message is reconstructed.

With reference to the OSI standard, the communication unit in a network is the packet. Packets are in turn divided into frames. The beginning and the end of each frame are usually determined by delimitation characters. The frames are in turn divided into information and control frames. The information frames transport the data relative to the message that is to be transmitted throughout the network, while the control frames deal with the regulating modes of said transport, i.e. the flow control and the starting of the error recovery actions. Both the information and the control frames contain a header portion identifying the frame type and a body portion which is typical of the frame itself.

The information frame structure will be described with reference to figure 2. In the upper portion of said figure, the generic structure of a OSI layer 2 packet is schematically described, thus comprising both information frames 1 and control frames 2. A single information frame (OSI layer 3) is constituted by a header portion 3, containing the identification that the frame is an information frame, and by a body portion 4. The body portion (OSI layers 4-7) contains the real message 5, together with a plurality of fields 6, typical of the particular application syntax used, illustrated by way of example in the figure with the characters C1, C2 and C3. The application syntax is the information relative to the number of fields contained within the plurality 6, to the meaning of each of said fields and to the data contained therein.

The OSI model schematically described up to this point is just a conceptual model. A typical protocol normally adopted is for example the TCP/IP (Transmission Control Protocol and Internet Protocol). Said protocol, just like other communication protocols adopted, can be explained with reference to the layers structure of the OSI model. In fact, in each of said protocols, a certain source layer will divide the data it receives from an upper layer adding to said data a header and/or a tail and will forward all this to a lower layer. On the destination side the opposite operations will occur.

With reference to the following figure 3, a schematic view is shown of the type of data used on local communication networks with reference to the TCP/IP protocol carrying the HTTP application service (Internet browsing).

The Ethernet Layer substantially includes four kinds of fields:

- a destination network card address field 101;
- a source network card address field 102;
- a communication protocol field 103, in this case indicative of the carried IP protocol and of the length of the information portion; and
- an information field 104, i.e. containing the Ethernet layer data, i.e. the entire structure of the carried IP protocol.

The IP Layer (encapsulated in the Ethernet layer) substantially includes six types of fields:

- a series of control fields 105 identifying the version, the length, the transmission options, the filler etc.;
- a communication protocol field 106, in this case indicative of the TCP protocol;
- an IP destination address field 107, i.e. of the IP address of the packet receiver;
- an IP source address field 108, i.e. of the IP address of the packet sender; and
- an information field 109, i.e. containing all the IP layer data, i.e. the entire structure of the carried TCP protocol.

The TCP layer (encapsulated in the IP layer) includes four types of fields:

- a source port field 110, indicating the TCP service port used by the packet sender;
- a destination port field 111, indicating the TCP service port used by the packet receiver;
- a series of control fields 112 identifying the packet ID, the working window, the crc, various options etc.; and
- an information field 113, i.e. containing the TCP layer data, i.e. the entire structure of the carried HTTP application service, i.e. the HTTP language commands and, in its information part, the HTML language commands.

Monitoring systems for the data transmitted between a sender node and a destination node are already known. However, said systems can only analyze the OSI layers 2 (data link) and 3 (network). The monitoring and the successive interpretation of the data at said layers allow only the monitoring of anomalies in the

exchange protocol among the various components of a network data transmission system.

Therefore, a typical disadvantage of said prior art systems is their incapability of decoding the application piece of information transported on the network, i.e. the piece of information related to the layers 4-7 of the OSI standard.

In the following figures 4 to 8B, the structure and the operation of an apparatus for monitoring and interpretation of application protocols will be described in detail.

Reference will now be made to figure 4, showing a block diagram of the apparatus. First of all, in said figure a source node 7 and a destination node 8 are shown, terminals of the network portion in which the data are monitored and interpreted. Throughout the connection between said two nodes, schematically illustrated by arrows F3, F4, F5, F6 and by the transmission medium 23, data relative to plural communications between a first set of source processors (not shown in the figure) upstream of the source node 7 and a second set of destination processors (not shown in the figure) downstream of the destination node 8 travel bidirectionally.

Said data are monitored by means of a data monitoring device 9. Several are the monitoring devices known on the market; for instance, concerning networks based on Ethernet technology, the Fast Etherlink XL™ card produced by the company 3Com™ can be mentioned. As for networks based on X.25 technology, e.g. the S508 card produced by the Canadian company Sangoma™ can be mentioned. Such card can operate with different OSI layer 1 (physical layer) standards such as, for example the RS232 (or V.24) standard and the RS422 (or V.35) standard. The OSI layer 2 (data link) standards together with said card can operate are, for instance, the HDLC standard, or the X.25 standard, contained therein. Anyway, the kind of data monitoring device 9 to be selected for the purposes of the present invention can vary depending on which OSI layers 1 or 2 standards one needs to operate. In fact, it will be possible to use monitoring devices working with implementation standards different from the OSI layer 2, such as for example "Frame Relay" or SDLC or also BSC and the like. Said devices are well known to the person skilled in the art and will not be here described in detail.

The monitoring occurs "transparently" by means of two parallel connectors 10 and 11, schematically illustrated in the figure, allowing the monitoring of the data coming respectively from the source node 7 and from the destination node 8. The monitoring device 9, shown by the dashed block in the figure, includes a source data receiver 12, a destination data receiver 13 and a connection interface 14.

The source data receiver 12 allows the reception of the data coming from the source node 7 only, as it is schematically indicated with the arrow F7; on the other hand, the destination data receiver 13 allows the reception of the data coming from destination node 8 only as schematically indicated with the arrow F8. The data received in this manner are transmitted to the connection interface 14, as it is indicated by the arrows F9 and F10.

Each data packet situated at a layer corresponding to the OSI layer 2 read by the monitoring unit 9 is forwarded to a control unit 15, as indicated by arrow F11. The control unit 15 will be described in detail later. To each of said packets a reading time is associated by means of a dating unit 16, represented outside the control unit 15 for ease of description and therewith connected as indicated by arrow F12. Such dating unit 16 can be any absolute time device on the market, in particular a radio or a satellite one. In a preferred embodiment of the present invention a radio controlled digital clock adjusted on the CET (Central European Time) broadcast by geostationary satellite was used.

Further to the association of the reading time by means of the dating unit 16, the control unit 15 discriminates the single frame so as to reconstruct the right logic/temporal forwarding sequence of the frames that, as it is known, does not always coincide with the received sequence: in fact, due to the routing techniques on telecommunications networks, it is possible for a forwarded sequence of the "ABC" type to be received in each of its possible permutations, i.e. "ABC", "ACB", "BAC", "BCA", "CAB", "CBA". Therefore, the control unit 15 discriminates the information frames from the control frames. For example, if transmission of the information occurs in the HDLC format, the last bit of the header portion of the information frame is 0 whereas the last bit of the header portion of a control frame is 1. Therefore, inside the control unit 15 there are means, not described in the figure, discriminating said last bit, e.g. a firmware contained in a ROM. In any case, no matter which data transmission code is used, the modes discriminating a control frame from a information frames will always be known. Therefore, it will always be possible to provide means for said discrimination. Such discrimination thus allows the storage of the single information frame deprived of the header portion and comprising the body portion only, thus containing the information which is typical of the particular application syntax used, together with the message to be transmitted.

The data incorporating the monitoring time and divided into information frames and control frames are stored inside a discriminated data storing unit 17, bidirectionally connected to the control unit 15 as indicated by arrow F13.

There is also a predetermined data storing unit 18, bidirectionally connected to the control unit 15. Said predetermined data represent possible interpretations of the information or control frames contained in the discriminated data storing unit 17. Their use will be explained herebelow with reference to the following figures. The connection between the predetermined data storing unit 18 and the control unit 15 is indicated by arrow F14.

Reference will now be made to figure 5, showing a flow chart indicating the operations executed by the control unit 15 on the information frames stored in the discriminated data storing unit 17. The access to such information frame is intended to be selectively regulated by authorizations and privileges management systems such as passwords, encryption and decryption codes, badge readers and the like given to qualified users, depending on the cases.

A first step S1 indicates the reading of the various packets by the monitoring unit 3. A second step S2 indicates the previously described discrimination operated by the control unit 15 between the information frames and the control frames, together with the association of the monitoring time.

On the non-application low layer control frames, whose use is marginal for the purposes of the present invention, a statistic processing might also be provided, operated in the step S3. Said processing is not described in detail at the moment; the modes by which it occurs will turn out to be clear at the end of the present description. The final result of such processing will provide a list of the control frames, reporting also the number of occurrences for each of said frames.

As for the information frames, the flow proceeds to a step S4 in which the single information frames are reconstructed according to their specific application syntax. To the purposes of said reconstruction, the application syntax structures of the single information frames must be known. In fact, they are contained inside the predetermined data storing unit 18 described with reference to the previous figure 3. Said unit 18 contains, for example in a text file, a formal abstract description for possible interpretations of the information or control frames. Said data represent the modes according to which the body portion of a single information frame can be structured, for example the machine transmission code (i.e. related to an information frame forwarded by the source or the destination), the number of the channel (i.e. related to a specific processor upstream of the source node or to a specific processor downstream of the destination node), protocol numbers, data processing numbers etc. Said unit 18 can of course contain the syntax

of several application protocols of the information frames that are to be reconstructed in that moment.

A reconstruction of the information frames one by one is obtained by a sequential comparison of the body portion of each information frame with each one of the abstract models in the unit 18.

Further to this, the different application sequences occurred between a determined source processor and a determined destination processor can be reconstructed, i.e. ordered according to time and kind of communication. Throughout the present description, for application sequence will be intended the whole of the information frames exchanged between a determined source processor and a determined destination processor during a single communication. The application sequence ordered in step S5 will contain the single information frames ordered according to a time criterion only and not also to a logic one. Ordering by time will be possible through the time association occurred in the previous step S2.

To give also a logical ordering of the data inside a specific application sequence, the presence of a group of application rules directing the data exchange between source and destination can be useful, although not necessary. Said application rules, typical of the particular kind of conversation between a determined source processor and a determined destination processor, must be predetermined and as such, they as well are collected in the predetermined data storing unit 18. Said application rules are a series of possible interpretations of the information frames sequences contained in the discriminated data storing unit 17.

An example of such application rules is given by table 1 herebelow, wherein reference is made to a communication between a source representing a student (client) wanting to enroll to university via terminal, and a destination (server) representing the university where the student wants to enroll.

TABLE 1

1: AS ? FDB 15 AS ? FDB 5 AS ? FDB 0
The enrollment booking was regularly acquired
2: AS ? FDB 13 AS ? FDB 0
The client position is not regular
.....
.....
.....

Every row of said table is an application rule, indicating i.e. a possible data exchange application sequence between source and destination. The meaning of

each application sequence is illustrated herebelow. For example, the first row indicates the following sequence of information frames:

- the source (AS) interrogates (?) the destination;
- the destination (FDB) answers with the activity number 15;
- the source (AS) interrogates (?) again the destination;
- the destination (FDB) answers with the activity number 5;
- the source (AS) interrogates (?) the destination; and
- the destination (FDB) answers with the activity number 0.

The result obtained at the end of this conversation is that the booking for the university enrollment is regularly acquired.

The structure of Table 1 is a mere example and it could also be illustrated with a tree structure having a number of branches depending on the number of application sequences provided. Every path down to one of the tree leaves would illustrate a particular application sequence, i.e. a particular conversation between source and destination, i.e. a particular information frame sequence between source and destination.

The number of application rules can be anyone. The larger the number of application rules provided, the bigger the chance to associate each of the application sequences temporally reconstructed in the step S5 with a well defined logic meaning, found by comparison with a particular application rule contained in the predetermined data storing unit 18 in figure 3. Therefore, in this manner it will be possible to verify the correctness or the anomaly of the particular application sequence that is being compared in that moment.

In the step S6 in figure 5 the control unit 15 verifies first of all whether such application rules be available or less. Supposing that said application rules are known, the flow can proceed either toward a step S8 or toward a step S9, depending on what was chosen in the step S7. The step S8 allows a simple classification of the application sequences. In fact, each application sequence is classified as belonging to a particular path among the various possible paths inside the application rules tree. The step S8 will be explained in greater detail with reference to the following figure 6.

On the other hand, in the step S9 the logical path of all the application sequences monitored by the apparatus in a predetermined time interval is reconstructed. Said step S9 will be described in greater detail with reference to the following figure 7.

The apparatus according to the present invention allows a reconstruction of the logical path of the application sequences also if a series of application rules is not provided. In this event, the flow proceeds to a step S10, that will also be described later.

Reference will now be made to figure 6, which provides a more detailed explanation of what previously described with reference to step S8 of figure 5. In a first step S11 the single application sequence, object of the comparison, is selected. In a successive step S12, the elements which are characterizing for comparison purposes are selected inside the selected application sequence.

In the example of the enrollment to university previously described in table 1 said characterizing elements might be: the identification number of the source processor, the identification number of the user who required the enrollment operation, the data provided by the source and the data provided by the destination.

In the step S13 the characterizing elements of the considered application sequence are compared with one of the application rules of the above described table 1 searching for a possible correspondence. If such a correspondence is found, the flow proceeds to a step S14 wherein said correspondence is reported and will have to be taken into consideration in the results of the interpretation. Then the flow selects another sequence and executes again the step S11. If the correspondence at the step S13 is not found, the control unit 15 goes in step S15 to a subsequent rule and if (step S16) there are still rules allowing a comparison the control unit executes once again the comparison of step S13. If no further rules are found, the control unit reports an anomaly in the step S17. Such an anomaly might alternatively mean:

- either a kind of sequence which should not have occurred (a real anomaly); or

- a kind of sequence not inserted by mistake inside the application rules tree.

In each of said events, finding such an anomaly is certainly useful for the certification of the kinds of application sequences occurred in the network portion under examination.

Reference will now be made to the following figure 6 which gives a more detailed explanation of what described in the step S9 in figure 5.

The steps S18 and S19 select respectively the single application sequence and the characterizing elements of the same, similarly to what described with reference to the previous figure 5. The step S20 is to indicate the comparison between the application sequence and the preset application rules contained inside

the predetermined data storing unit 18. If a correspondence is found, the flow proceeds to a step S21 wherein the correspondence found is taken into consideration through the update of the related statistic fields. Steps S18-S20 will be subsequently repeated, until the end of the sequences to be classified. If no correspondence is found, the application sequence to be classified is new; it can be an anomaly or simply an unexpected sequence. In this event, the flow proceeds to a step S22 wherein the statistic fields related to that specific sequence are initialized. Furthermore, the new sequence will be inserted in the list of the preset sequences that are to be used for the comparison in the step S20. This is also indicated by the double pointing of the arrow F14 in the previous figure 4. Said particular sequences, i.e. the possible anomalies, can be evidenced in a particular manner to be recognized as such. Further to this, also in this case the steps S18-S20 are repeated until the end of the sequences to be classified. In particular, besides the number of crossings for each tree branch, it is also possible to monitor uncrossed branches.

In case there is no preset sequence of application rules, it will always be possible for the control unit to reconstruct the communication applications occurred in the network portion under control (step S9 in figure 5). In this event each analyzed application sequence will not be compared with the preset sequences, but with the previously analyzed sequences. Therefore, the tree structure containing statistic information will be reconstructed by reciprocal comparison of the body portion of the information frames. Also in this case, a tree will be constructed and it will be possible to know the number of crossings for each branch. Obviously, in this case it will not be possible to monitor the uncrossed branches as there will not be a prior knowledge of the existence of said branches.

Reference will now be made to figures 8A and 8B showing respectively an example of an information frame and an example of a tree structure containing statistic information obtained by means of the apparatus according to the present invention.

In figure 8A it is possible to notice four different fields: a first field 19 indicating the name of the source or destination processor; a second field 20 indicating the number of connections in the monitored time interval, a third field 21 indicating the average time length of each connection, counted for example in milliseconds, and a fourth field 22 indicating the code of the activity executed.

Figure 8B indicates the reconstructed tree. A first element E1 in the tree indicates that AS (source) connected 20 times, with an average connection time of 0 milliseconds (simple opening of the connection with the destination) and

executed the activity with the code 0. A second element E2, E1's only "son", indicates that in all those 20 connections FDB (destination) answered with the activity having the code 20, with an average connection time of 20 milliseconds. There were two manners of proceeding. AS answered for 18 times (element E3) with the activity 0 and twice (element E4) with the activity 1. The tree proceeds with other elements, whose meaning is now clarified by the context. The tree herewith disclosed is the result of the logical ordering operated in the steps S9 or S10 in figure 5.

It is to be noted that the monitoring of the contents in the fields 19 and 22 of each element was operated in the step S4 in figure 5. The monitoring of the connections among the various elements, i.e. the fact that the element E2 is E1's "son" and that the elements E3 and E4 are E2's "sons" was operated either in the step S9 or in the step S10 in figure 5.

Having ended the detailed description of an apparatus for monitoring and interpretation of network application protocols, herebelow the structure and the operation of the remaining components of the network access monitoring device according to the present invention will be described in detail.

The preferred connection mode of said device is a series connection, on Ethernet networks for 10 Mbits (connectors rj58 and rj45) and for 100 Mbits (rj45) or more.

The OSI layer 2 supported protocols will be all the protocols encapsulated in Ethernet, like 802.3, DOD IP, ARP etc.

The OSI layer 3 supported protocols will be all the protocols encapsulated in the various OSI layer 2 protocols, like TCP/IP, UDP/IP, Netbios/IEEE 802.3, SNA/IEEE 802.3 etc.

First of all, reference will be made to figure 9, showing a block diagram of the access control device according to the present invention. The various blocks in figure 9 will be described herebelow one at a time.

Element 201:

It is the element storing the pattern recognition rules. The archive of the recognition rules is created reading a file or, e.g. directly typing in the rules through the keyboard.

Firstly, it should be assumed that said recognition rules are indicated as <data type>/<data value> pairs.

For instance, a recognition pattern of an Internet browsing request by a client with the address 192.23.40.1 to a web server of address 210.20.20.6 has the following structure:

(ETH_PROT, IP),
(IP_SRC_ADDR, 192.23.40.1),
(IP_DST_ADDR, 210.20.20.6),
(TCP_DST_PORT, HTTP)

wherein:

the first pair (ETH_PROT, IP) indicates that the protocol contained in the Ethernet layer must be the IP protocol;

the second pair (IP_SRC_ADDR, 192.23.40.1) indicates that the IP address of the packet sender must be the one indicated;

the third pair (IP_DST_ADDR, 210.20.20.6) indicates that the IP address of the packet receiver must be the one indicated; and

the fourth pair (TCP_DST_PORT, HTTP) indicates that the TCP service used is the HTTP (web) one.

The identification numbers on the right of the pairs can also assume values that are not predetermined, for instance if it is desirable to identify all the addresses of a subnetwork. In this case the address of the previous example can be expressed as 210.20.20.* where the symbol * (asterisk) indicates a joker value, i.e. all the possible values that can be in that position. In the same pair two or more asterisks may appear: e.g., 210.*.20.*, thus indicating a set of 65536 (or more) different addresses. Other accepted configurations are e.g.: 2*.20.20.* indicating all the addresses beginning with 2 and ending with a subaddress comprised between 0 and 255 (in this case a total of 100*256=25600 different addresses).

A further example of a recognition pattern for the IBM NetBios protocol between two processors is the following:

(ETH_PROT, IEEE802),
(IEEE802_DST_SAP, IBM_NETBIOS)

When wishing to force the recognition of the network cards involved in the NetBios communication (6 bytes, including the card issuer code and the card number), the pattern becomes:

(ETH_SRC_ADDR, 0xFF45DE782201),
(ETH_DST_ADDR, 0xF237C811000F),
(ETH_PROT, IEEE802),
(IEEE802_DST_SAP, IBM_NETBIOS).

Element 202:

It is the pattern compiler, consisting of a conversion element for converting the rules contained in 201 into a set of sequences of numerical

identificators and consisting of a compression element for compressing the identificators thus obtained.

i) Conversion element

The recognition rules appearing as <data type>/<data value> pairs are converted into sequences of numerical identificators, constituting the basis for the recognition of the frames read from the network.

For instance, given the rule

(ETH_PROT, IP),

(IP_SRC_ADDR, 228.186.33.90),

(IP_DST_ADDR, 41.240.227.149),

(TCP_DST_PORT, HTTP)

it follows that:

a) the first pair (ETH_PROT, IP) is converted into two hexadecimal data pairs (in which the 0x prefix indicates that the subsequent value is represented in hexadecimal):

0x0C 0x0800

0x49 0x06

wherein:

- the first row contains two values, 0C and 0800. The digit farther left of the first value (0) identifies an Ethernet frame. The second digit of the first value (C) indicates the position inside the frame(13th byte, considering the first one to be in position 0). The second value (0800) is the identification code of the IP protocol when contained in an Ethernet frame; and

- the second row contains two values, 49 and 06. The digit farther left of the second value (4) identifies an IP net. The second digit of the first value (9) indicates the position inside the net. The second value (06) identifies the TCP protocol contained in IP.

b) the second pair (IP_SRC_ADDR, 228.186.33.90) is converted into four hexadecimal data pairs:

0x4C 0xe4

0x4D 0xba

0x4E 0x21

0x4F 0x5a

wherein each pair indicates respectively the IP frame(4), the position (from C to F) and the value of each single element constituting the source address: in fact e4 in hexadecimal corresponds to 228 in decimal, ba in hexadecimal corresponds to 186

in decimal, 21 in hexadecimal corresponds to 33 in decimal and 5a in hexadecimal corresponds to 90 in decimal.

c) the third pair (IP_DST_ADDR, 41.240.227.149) is converted into four hexadecimal data pairs:

0x410 0x29

0x411 0xF0

0x412 0xE3

0x413 0x95

wherein each pair indicates respectively the IP frame(4), the position (from 10 to 13) and the value of each single element constituting the destination address: in fact 29 in hexadecimal corresponds to 41 in decimal, F0 in hexadecimal corresponds to 240 in decimal, E3 in hexadecimal corresponds to 227 in decimal and 95 in hexadecimal corresponds to 149 in decimal.

d) the fourth pair (TCP_DST_PORT, HTTP) is converted into a pair of hexadecimal data:

0x82 0x0080

in which the digit farther left of the first value (8) indicates the TCP frame, the second digit of the first value (2) indicates the position inside said frame (the third starting from zero) whereas the second value 0080 indicates the HTTP service (the one used by web applications).

Therefore, starting from rule

(ETH_PROT, IP),

(IP_SRC_ADDR, 228.186.33.90),

(IP_DST_ADDR, 41.240.227.149),

(TCP_DST_PORT, HTTP)

the sequence

0x0C 0x0800, 0x49 0x06, 0x4C 0xe4, 0x4D 0xba, 0x4E 0x21, 0x4F 0x5a, 0x410 0x29, 0x411 0xf0, 0x412 0xe3, 0x413 0x95, 0x82 0x0080

is obtained.

It is intended that all the conversions hereto described are made possible through a sequential comparison of each of the <data type>/<data value> pairs with a table storing all possible <data type>/<data value> pairs together with the corresponding hexadecimal data pair.

Actually, a more extended form can be used for the rules thus defined, capable of being semantically represented by the <object>/<action> pair. The <object> field indicates the set of properties (including the value) assumed by the

element currently under examination, whereas the <action> field expresses the actions that are to be executed after having recognized said object in the communication frame.

For instance, in the hexadecimal pairs of the <data type>/<data value> kind it is apparent how the <data type> field contains a double information, i.e. both the protocol (or the frame type) to which reference is made, and the position inside said protocol.

In the event of complex application protocols, the monitored frames are usually represented by means of a language of the LL(1) type (i.e., according to the definition of Chomsky, a language having no control structures and with no limitations for the definition of the interpretation processes of the information structures). In this event, the <action> field will make reference to a minimal set of basic instructions reported herebelow:

- Push
<value>
<variable>
<reading position>
<value at the reading position>
- Pop
<variable>
<reading position>
<in the reading position>
- And
- Mul
- Add
- Equal
- Next
- F_send_all
- F_dynamic

Herebelow a schematic outline of the meaning of said basic instructions is provided for sake of completeness.

- Push <value> inserts a value in the stack dedicated to the recognition process under way, for instance: PUSH(35), the value 35 is inserted in the stack;

- Push <variable> inserts the content of a variable in the stack dedicated to the recognition process under way, for instance: PUSH(v12), if the value of the "v12" variable is 8, then 8 is inserted in the stack;

- Push <reading position> inserts in the stack dedicated to the recognition process under way the position of the value currently read in the input stream, for instance PUSH(pos) if the value of pos, a variable indicating the reading position, is 5, then 5 is inserted in the stack;

- Push <value at the reading position> inserts the value read in the input stream under recognition in the "reading position" in the stack dedicated to the recognition process under way, e.g. PUSH(v_pos), if the value of pos, a variable indicating the reading position, is 5 and if at position 5 of the input stream there is the value 30, then 30 is inserted in the stack;

- Pop <variable> inserts the stack head in the "variable" variable e.g. POP(v3), if the value 10 was inserted in the stack head, meaning that the last operation performed with the stack was e.g. push(10), then the value 10 goes into the "v3" variable;

- Pop <reading position> inserts the stack head in the variable indicating the successive position to be read in the input stream, e.g. POP(pos), if in the stack head the value 10 has been inserted, then the next element that will be read by the input stream will be the one in position 10;

- Pop <in the reading position> inserts the stack head in the position indicated by the variable indicating the next position to be read in the input stream, e.g. POP(v_pos), if the value 10 has been inserted in the stack head, the value of the next element that will be read by the input stream will be 10;

- And, Mul, Add, Or, Sub are all logical and arithmetical operations. The operation is performed on the values contained in the first two stack positions, the result becomes the stack head and the two used values are removed from the stack; example: the logical arithmetical operations follow the reversed Polish notation (RPN). It is now supposed to have to execute the operation 10*30 to be executed: the entailed program will be:

PUSH(10)
PUSH(30)
MUL
now, in the stack head there is 300=30*10.

- Equal <value>, Equal <variable>, Equal <reading position>, Equal <value at the reading position> verifies whether in the stack head there is a value equal to

the one forwarded as a parameter. The result (0 if different, 1 if equal) is inserted in the stack head;

- f_send_all is a function that, when operated, reports the entire input stream to the output;

- Next <value>, Next <variable> increments of the value contained in "value" or of the value contained in "variable" the variable indicating the input stream position from which the next value is to be read; lastly

- f_dynamic("name") operates the "name" function connected to the coordinating element through dynamic connection mechanisms (as DLL in Windows or shared_libraries in UNIX, or RPC/DCE mechanisms, ...) forwarding thereto the values contained in the stack as parameters.

A possible implementation syntax (adopted from C language) of the set of the <item>/<action> pairs can be the following one:

```
typedef struct _item {
        unsigned char object;
        unsigned long int action;
} Item;

typedef struct _record {
    int num_of_items;
    Item * items;
} Record;

Record * input_second_step;
```

wherein:

- the "object" field is expressed as a single byte ("unsigned char"). Such a choice does not entail limitations, since an entire value (2 to 4 bytes long) can be considered as a sequence of bytes and therefore it can be processed one byte at the time; and

- the field "action" is expressed as "unsigned long int". Hence, it can represent both a number (compatible with the first notation) and a pointer to a structure or to a set of functions (compatible with the second notation).

Usually the number of different sequences is very high. By way of example, taking into consideration exclusively the TCP-IP protocol, for a relatively low number of 1000 "clients" (i.e. of processors using application services made

available by other processors) and of 10 "servers" (i.e. of processors providing application services to the clients) and of an average of 10 application services per "server" (as e.g. FTP, TELNET, HTTP, MAIL, NFS, TIME, DNS), in order to discriminate all the possible "pairings" among client-server-service, rules indicating 1000*10*10=100000 different pattern sequences in the communication frames have to be determined.

This number, already well above the dimension deemed acceptable for the internal addressing tables of the routers and of the commercial firewalls, increases very rapidly when rules operating not merely at the level of a control portion of the communication protocols, but operating on the level of the data portion as well are determined, as is the case in the present invention.

The above defined language of rules allows to write rules allowing the identification of elements of the data portion of the communication protocol: in fact, if not merely the "identification" of a "client" is desirable, but also when he tries to access a specific WEB page from a network (which is possible by means of the present invention), it is not enough to operate at a level of the communication protocol control portion (only the fact that a command was sent at the level of an HTTP service would be recognized) being it necessary instead to operate at a level of the TCP-IP protocol data portion in order to identify the particular string determining access to the WEB page requested by the client.

ii) Compression element of the set of sequences obtained in a direct access data structure

Said second element of the pattern compiler 202 allows a construction of the compression data structure ensuring a constant access time (i.e., regardless of the number of sequences) and an optimal memory occupation (i.e., equal to the amount of memory required to store sequences in a non structured way multiplied for a constant value) for recognition of the sequences stored in said structure in the communication frames that are readable from a network. Moreover it is possible to update such a data structure in a number of steps proportional to N*log(N), where N is the number of new sequences to be inserted.

In particular, reference will be made to the articles:

a) "Time Optimal Digraph Browsing on a Sparse Representation", Mathematics Department Tech. Report, Tor Vergata University of Rome 8/97, 1997 by M. Talamo and P. Vocca;

b) "Optimal Bounds on Complexity of Sparse Partial Orders", Mathematics Department Tech. Report, Tor Vergata University of Rome, 9/97, 1997 by M. Talamo and P. Vocca;

c) "Optimal Digraph Search on a Compressed Representation", Mathematics Department Tech. Report, Tor Vergata University of Rome, 11/98, 1998 by M. Talamo and P. Vocca; and

d) "Compact Implicit Representation of Graphs", WG98 proceedings, June 1998 by M. Talamo and P. Vocca.

In said articles data structures allowing a constant time access, i.e. regardless of the number of data represented by them, are described.

The algorithm for obtaining said data structures is applied to access structures of the "bipartite graph" kind, e.g. as the one represented in figure 10A. In such a graph the nodes can be separated into two separate subgroups (from A to E and from 0 to 4 in figure), in such a way that each node belonging to a first subset can be connected only with nodes belonging to the second subset and vice versa. With reference to figure 10A, node A is connected with node 0 and node 2, node B is connected with node 0 and node 2, node C is connected with node 1 and node 4, node D is connected with node 3, and node E is connected with node 3.

Such connections can be expressed by means of a bidimensional matrix of the kind reported in figure 10B, where with the symbol x the connections active between lines and columns have been represented. Therefore, it can be concluded that the bipartite graphs are equivalent to the bidimensional matrices and that therefore the constant time accessibility results obtained with reference to the above cited article can also apply to structures such as bidimensional matrices.

Therefore, the compression element will compress the sequences obtained through the conversion element, and will generate a variety of bidimensional matrices indicating such sequences.

* * *

The algorithm by which the compression element operates, described herebelow, (from STEP 1 to STEP 11) is intended to be implemented in any suitable programming language (e.g. in C language) and stored in a ROM.

The input to the algorithm consists in the sequence of numerical identificators (records) of a preset variable length.

By way of example, together with the various algorithm steps a complete compilation cycle for a specific practical case will be reported, in order to fully describe the operation manners of the algorithm itself. Accordingly, reference

to communication structures of the Ethernet kind will be made again. Obviously, the operation of the control device according to the present invention remains unaltered even in the event the apparatus for monitoring and interpretation does not provide Ethernet frames monitored on the network, but directly provides instead TCP/IP communications or anyhow very long data streams.

STEP 1 (specification of predetermined rules, see also figure 11):

It is supposed to have to manage and coordinate Ethernet communication frames by means of the following connection diagrams:

connection a) 132.147.200.10 can connect with 132.147.160.1 only for the service:

- WWW, service TCP 80.

connection b) 132.147.200.10 can connect with 132.147.160.2 for the services only:

- SMTP, service TCP 25;
- NETBIOS, services TCP 137, 138 and 139.

connection c) 132.147.200.20 can connect with 132.147.160.1 only for the services:

- FTP, services TCP 20 and 21;
- TELNET, service TCP 23.

connection d) 132.147.200.20 can connect with 132.147.160.2 only for the services:

- SMTP, service TCP 25;
- WWW, service TCP 80.

connection e) 132.147.200.20 can connect with 132.147.160.3 only for the services:

- WWW, service TCP 80;
- SNMP, services TCP 161 and 162;
- NFS, service TCP 2049;
- TELNET, service TCP 23.

Further, all the communications of the ARP (Ethernet layer protocol) and ICMP (IP-layer protocol) kind will have to be accepted.

STEP 2 (conversion of rules in a set of sequences):

A set of 17 records (in which each record consists of a set of <item>/<action> pairs) is obtained based on said connection diagram. In particular, record 1 represents connection a), records 2 to 5 represent connection b), records 6 to 8 represent

connection c), records 9 to 10 represent connection d), records 11 to 15 represent connection e), record 16 represents the Ethernet ARP protocol, and lastly record 17 represents the ICMP protocol in IP.

Connection a)	
RECORD 1	
0x08, 0x000C	
0x00, 0x000D	IP protocol in Ethernet
0x06, 0x4009	TCP protocol in IP
0x84, 0x400C	
0x93, 0x400D	
0xC8, 0x400E	
0x0A, 0x400F	132.147.200.10
0x84, 0x4010	
0x93, 0x4011	
0xA0, 0x4012	
0x01, 0x4013	132.147.160.1
0x00, 0x8002	
0x50, 0x8003	WWW 80
Connection b)	
RECORD 2	
0x08, 0x000C	
0x00, 0x000D	IP protocol in Ethernet
0x06, 0x4009	TCP protocol in IP
0x84, 0x400C	
0x93, 0x400D	
0xC8, 0x400E	
0x0A, 0x400F	132.147.200.10
0x84, 0x4010	
0x93, 0x4011	
0xA0, 0x4012	
0x02, 0x4013	132.147.160.2
0x00, 0x8002	
0x19, 0x8003	SMTP 25

RECORD 3

0x08, 0x000C	
0x00, 0x000D	IP protocol in Ethernet
0x06, 0x4009	TCP protocol in IP
0x84, 0x400C	
0x93, 0x400D	
0xC8, 0x400E	
0x0A, 0x400F	132.147.200.10
0x84, 0x4010	
0x93, 0x4011	
0xA0, 0x4012	
0x02, 0x4013	132.147.160.2
0x00, 0x8002	
0x89, 0x8003	NETBIOS 137

RECORD 4

0x08, 0x000C	
0x00, 0x000D	IP protocol in Ethernet
0x06, 0x4009	TCP protocol in IP
0x84, 0x400C	
0x93, 0x400D	
0xC8, 0x400E	
0x0A, 0x400F	132.147.200.10
0x84, 0x4010	
0x93, 0x4011	
0xA0, 0x4012	
0x02, 0x4013	132.147.160.2
0x00, 0x8002	
0x8A, 0x8003	NETBIOS 138

RECORD 5

0x08, 0x000C	
0x00, 0x000D	IP protocol in Ethernet
0x06, 0x4009	TCP protocol in IP
0x84, 0x400C	
0x93, 0x400D	

0xC8, 0x400E	
0x0A, 0x400F	132.147.200.10
0x84, 0x4010	
0x93, 0x4011	
0xA0, 0x4012	
0x02, 0x4013	132.147.160.2
0x00, 0x8002	
0x8B, 0x8003	NETBIOS 139

Connection c)

RECORD 6

0x08, 0x000C	
0x00, 0x000D	IP protocol in Ethernet
0x06, 0x4009	TCP protocol in IP
0x84, 0x400C	
0x93, 0x400D	
0xC8, 0x400E	
0x14, 0x400F	132.147.200.20
0x84, 0x4010	
0x93, 0x4011	
0xA0, 0x4012	
0x01, 0x4013	132.147.160.1
0x00, 0x8002	
0x14, 0x8003	FTP 20

RECORD 7

0x08, 0x000C	
0x00, 0x000D	IP protocol in Ethernet
0x06, 0x4009	TCP protocol in IP
0x84, 0x400C	
0x93, 0x400D	
0xC8, 0x400E	
0x14, 0x400F	132.147.200.20
0x84, 0x4010	
0x93, 0x4011	
0xA0, 0x4012	

0x01, 0x4013	132.147.160.1
0x00, 0x8002	
0x15, 0x8003	FTP 21

RECORD 8

0x08, 0x000C	
0x00, 0x000D	IP protocol in Ethernet
0x06, 0x4009	TCP protocol in IP
0x84, 0x400C	
0x93, 0x400D	
0xC8, 0x400E	
0x14, 0x400F	132.147.200.20
0x84, 0x4010	
0x93, 0x4011	
0xA0, 0x4012	
0x01, 0x4013	132.147.160.1
0x00, 0x8002	
0x17, 0x8003	TELNET 23

Connection d)

RECORD 9

0x08, 0x000C	
0x00, 0x000D	IP protocol in Ethernet
0x06, 0x4009	TCP protocol in IP
0x84, 0x400C	
0x93, 0x400D	
0xC8, 0x400E	
0x14, 0x400F	132.147.200.20
0x84, 0x4010	
0x93, 0x4011	
0xA0, 0x4012	
0x02, 0x4013	132.147.160.2
0x00, 0x8002	
0x19, 0x8003	SMTP 25

RECORD 10

0x08, 0x000C	
0x00, 0x000D	IP protocol in Ethernet
0x06, 0x4009	TCP protocol in IP
0x84, 0x400C	
0x93, 0x400D	
0xC8, 0x400E	
0x14, 0x400F	132.147.200.20
0x84, 0x4010	
0x93, 0x4011	
0xA0, 0x4012	
0x02, 0x4013	132.147.160.2
0x00, 0x8002	
0x50, 0x8003	WWW 80

Connection e)

RECORD 11

0x08, 0x000C	
0x00, 0x000D	Ethernet IP protocol
0x06, 0x4009	TCP/IP protocol
0x84, 0x400C	
0x93, 0x400D	
0xC8, 0x400E	
0x14, 0x400F	132.147.200.20
0x84, 0x4010	
0x93, 0x4011	
0xA0, 0x4012	
0x03, 0x4013	132.147.160.3
0x00, 0x8002	
0x50, 0x8003	WWW 80

RECORD 12

0x08, 0x000C	
0x00, 0x000D	IP protocol in Ethernet
0x06, 0x4009	TCP protocol in IP
0x84, 0x400C	

0x93, 0x400D	
0xC8, 0x400E	
0x14, 0x400F	132.147.200.20
0x84, 0x4010	
0x93, 0x4011	
0xA0, 0x4012	
0x03, 0x4013	132.147.160.3
0x00, 0x8002	
0xA1, 0x8003	SNMP 161

RECORD 13

0x08, 0x000C	
0x00, 0x000D	IP protocol in Ethernet
0x06, 0x4009	TCP protocol in IP
0x84, 0x400C	
0x93, 0x400D	
0xC8, 0x400E	
0x14, 0x400F	132.147.200.20
0x84, 0x4010	
0x93, 0x4011	
0xA0, 0x4012	
0x03, 0x4013	132.147.160.3
0x00, 0x8002	
0xA2, 0x8003	SNMP 162

RECORD 14

0x08, 0x000C	
0x00, 0x000D	IP protocol in Ethernet
0x06, 0x4009	TCP protocol in IP
0x84, 0x400C	
0x93, 0x400D	
0xC8, 0x400E	
0x14, 0x400F	132.147.200.20
0x84, 0x4010	
0x93, 0x4011	
0xA0, 0x4012	

0x03, 0x4013	132.147.160.3
0x08, 0x8002	
0x01, 0x8003	NFS 2049

RECORD 15

0x08, 0x000C	
0x00, 0x000D	IP protocol in Ethernet
0x06, 0x4009	TCP protocol in IP
0x84, 0x400C	
0x93, 0x400D	
0xC8, 0x400E	
0x14, 0x400F	132.147.200.20
0x84, 0x4010	
0x93, 0x4011	
0xA0, 0x4012	
0x03, 0x4013	132.147.160.3
0x00, 0x8002	
0x17, 0x8003	TELNET 23

and lastly

RECORD 16

0x08, 0x000C	
0x06, 0x000D	ARP protocol in Ethernet

RECORD 17

0x08, 0x000C	
0x00, 0x000D	IP protocol in Ethernet
0x01, 0x4009	ICMP protocol in IP

The structure thus obtained can be expressed in a matricial form, according to the representation in figure 12. It is to be noted that the various records can have different lengths. In fact, there are 15 records of length 13, 1 record of length 2 and 1 record of length 3.

STEP 3:

Set CONT = 0

STEP 4:

Column 0 and column CONT of the above reported sequence are taken, and a new sequence of records containing only 2 items (the one in column 0 and the one in column CONT) is created.

STEP 5:

Doubles are eliminated from this new sequence of records.

STEP 6:

Set ROW=0

STEP 7:

A weighted bipartite graph is created with the new record sequence, by inserting for each record:

- the value of the item in position 0 (upper node id);
- the value of the item in position CONT (lower node id);
- the action of the item in position CONT (as the first weight of the arc between the two nodes);
- ROW (as the second weight of the arc between the two nodes).

Further, for each pair of inserted nodes, the item value in the position 0 in the original record sequence is replaced with the new ROW value, and then ROW=ROW+1 is set.

STEP 8:

The bipartite graph thus obtained is converted into a bidimensional matrix and a vector by means of the basic algorithm of which at the above mentioned articles. Note however that the algorithm described herewith constitutes an extension of said basic algorithm, in particular concerning the previous step 7.

STEP 9:

The bidimensional matrix and the vector are stored.

STEP 10:

Set CONT=CONT+1

STEP 11:

If CONT is not equal to the maximum number of items of the records the step 4 is repeated, else the algorithm is ended.

* * *

The sequence of bidimensional matrices and vectors constitutes the compressed data structure that will be used for recognition of the input streams. Such structure is accessible in a direct manner.

Herebelow again reference to figure 9 will be made.

Element 203 (memory containing the compressed patterns):

Such element consists of the sequence of matrices resulting from the above mentioned compression algorithm. By virtue of the high compression rate of said algorithm, the dimension of this sequence of matrices is directly proportional to the number of active connections of the original matrix, being therefore directly storable in the central memory. In case of a high number of active connections (>100.000.000), said sequence of compressed matrices can be managed by means of mass storage devices.

Element 204 (pattern recognizer):

Such element allows the comparison between the application frames to be recognized, monitored by means of the element 205 and the direct access data structure stored in 203.

The element 204 is realized in a microchip, and it substantially consists of a software implementing a direct accessing technique on matrices, in order to access the matrices stored in 203.

Therefore, the acceptability or non acceptability of the frame read from the network can be recognized in a completely deterministic way.

* * *

In order to provide a detailed example of the operation of said recognizer, herebelow first of all the structure of the matrices stored in 203 is reported, using a syntax similar to that of the C language:

```
//Structure for a bidimensional Matrix and a vector
typedef struct _matrices_AB {
    unsigned long int row_a;    //Number of rows of the matrix
    unsigned long int col_a;    //Number of columns of the matrix
    unsigned long int col_b;    //Number of elements of the vector
    unsigned long int **mA;     //Matrix of Values
    Action ***mP;               //Matrix of Actions
```

```
        unsigned long int *mB;      //Vector
} mat_AB;

typedef struct _vec_matrices_AB {
        mat_AB * MAB;               //Set of matrices and vectors
        unsigned long int num_mab;//Number of matrices and vectors
} * Vec_mat_AB;
```

Five input records and the resulting matrices are reported herebelow. In such example the description of the records is performed by means of the <item>/<action> syntax hereto reported. The associated actions are extremely simplified (a single action per each recognition). Moreover, for sake of simplicity, the recognition is assumed to begin always from the first byte of the input stream.

RECORD 1
0x01 next(1)
0x03 next(1)
0x02 f_send_all

RECORD 2
0x01 next(1)
0x06 next(1)
0x04 f_send_all

RECORD 3
0x02 next(1)
0x07 next(1)
0x03 f_send_all

RECORD 4
0x01 next(1)
0x02 f_send_all

RECORD 5
0x05 next(1)
0x01 f_send_all

For next(1) the action of positioning on the successive byte in the data stream is intended. For f_send_all the action of forwarding of to the outside all the data stream is intended.

By means of the aforedescribed algorithm the following matricial structure is obtained:

1	MATRIX OF VALUES 0: [X] 1: [0] 2: [1] 3: [X] 4: [X] 5: [2]	MATRIX OF ACTIONS (1, 0) NEXT(1) (2, 0) NEXT(1) (5, 0) NEXT(1)	VECTOR B X 0 0 X X 0
2	MATRIX OF VALUES 0: [X] 1: [4] 2: [0] 3: [1] 4: [X] 5: [X] 6: [2] 7: [3]	MATRIX OF ACTIONS (1, 0) F_SEND_ALL (2, 0) F_SEND_ALL (3, 0) NEXT(1) (6, 0) NEXT(1) (7, 0) NEXT(1)	VECTOR B 0 0 0
3	MATRIX OF VALUES 0: [0 4] 1: [X X] 2: [1 X] 3: [3 X] 4: [2 X]	MATRIX OF ACTIONS (2, 0) F_SEND_ALL (3, 0) F_SEND_ALL (4, 0) F_SEND_ALL	VECTOR B 0 0 0 0 1

In order not to overburden the present description, the various steps from the records to the graphically described matricial structure (after all, simple applications of the above described algorithm) will not here be described in detail. Moreover, for sake of clarity the matrix of values has been represented as physically separated from the matrix of actions.

Instead, the comparison steps that are performed in order to recognize or not the monitored data streams will be described in detail. Said steps relate to the specific case of matricial structure hereto described.

1) EXAMPLE OF RECOGNITION IN THE EVENT THE STREAM IS IDENTICAL TO THE RECORD 1: 0x01 0x03 0x02

First read value is 01.

Being in an initial condition, it is used as an index of the matrix as well as of the vector.

The adopted Matrix/Vector pair is in the position 1 of the above reported list.

The row index of matrix A is determined by the element which has been read, i.e. Row A =01, i.e. the first row.

The column index of matrix A is determined by the value contained by vector B at the position corresponding to the element which has been read, i.e. Column A =B[0x01]=0, i.e. the 0-th column.

Therefore, the value reported in A[1, 0], i.e. 0 will be read. Said value is the successive index of vector B.

Next, the action reported in A [1, 0], i.e. the numeric value corresponding to the action next(1) will be read.

Therefore the aforesaid action will be executed, thus proceeding to the successive data stream value.

The successive value will be reached, using then the Matrix/Vector pair at position 2 of the above reported list.

Read value is 03.

The row index of Matrix A is determined by the element which has been read, i.e. Row A =03, i.e. the third row.

The column index of matrix A is determined by the value contained in vector B at the position corresponding to the value reported in A[1, 0] obtained in the previous step (i.e. 0). Column A =B[0]=0, i.e. the 0-th column.

Therefore, the value reported in A[3, 0] i.e. 1 will be read. Such value is the next index of vector B.

Then the action reported in A[3, 0], i.e. the numeric value corresponding to the action next(1) will be read.

Therefore, the aforesaid action will be executed, and the position shall be shifted to the next value of the data stream.

The subsequent value will be reached and the Matrix/Vector pair at position 3 of the above reported list will be used.

Read value is 02.

The row index of matrix A is determined by the element which has been read, i.e. Row A =02, i.e. the second row.

The column index of matrix A is determined by the value contained in vector B at the position corresponding to the value reported in A[3, 0] obtained in the previous step (i.e. 1). Column A =B[1]=0, i.e. the 0-th column.

Therefore, the value reported in A[2, 0], i.e. 1, will be read. Such value is the successive index of vector B.

Then the action reported in A[2, 0], i.e. the numeric value corresponding to the action f_send_all, will be read. This means that recognition has occurred.

2) EXAMPLE OF RECOGNITION IN THE EVENT THE STREAM DIFFERS FROM THE RECORDS: 0x04 0x01

First read value is 04.

Being in an initial condition, it is used as an index of the matrix as well as of the vector.

The adopted Matrix/Vector pair is in the position 1 of the above reported list.

The row index of matrix A is determined by the element which has been read, i.e. Row A =04, i.e. the fourth row.

The column index of matrix A is determined by the value contained by vector B at the position corresponding to the element which is read, i.e. Column A=B[04]=X. Therefore the stream is not recognized.

3) EXAMPLE OF RECOGNITION IN THE EVENT THE STREAM DIFFERS FROM THE RECORDS: 0x01 0x05 0x03

The first read value is 01.

Being in an initial condition, it is used as an index of the matrix as well as of the vector.

The adopted pair Matrix/Vector is in the position 1 of the above reported list.

The row index of matrix A is determined by the element which has been read, i.e. Row A =04, i.e. the fourth row.

The column index of matrix A is determined by the value contained by vector B at the position corresponding to the element which is read, i.e. Column A =B[0x01]=0, i.e. the 0-th column.

Therefore, the value reported in A[1, 0], i.e. 0 will be read. Said value is the successive index of vector B.

Next, the action reported in A[1, 0] i.e. the numeric value corresponding to the action next(1) will be read.

Therefore, the position shall be shifted to the next value and the Matrix/Vector pair which is at position 2 in the above reported list will be used.

Read value is 05.

The row index of Matrix A is determined by the element which has been read, i.e. Row A =05, i.e. the fifth row.

The column index of Matrix A is determined by the value contained in vector B at the position corresponding to the value reported in A[1, 0] obtained in the previous step. Column A =B[0]=0.

Thus, the value reported in A[5, 0], i.e. X will have to be read. Therefore, the stream is not recognized.

Hence, using a direct access technique on matrices, easily operable in a microchip, the pattern recognizer is able to recognize the acceptability or the non acceptability of the input stream in a completely deterministic manner, in a number of accesses to matrices and vectors equaling the number of elements that are recognized in the same stream.

* * *

Herebelow, reference will be again made to figure 9.

Element 205:

It is the component for monitoring and acquisition of the communication frames. By means of this apparatus, an example of which has already been described in detail with reference to the preceding figures from 4 to 8B, the data acquisition also at an application level is made possible, i.e. the piece of information related to the layers 4-7 of the OSI standard. Such apparatus will be able to accept commands as CONNECT, SEND, RECEIVE and CLOSE in the event high layer application protocols have to be managed and coordinated.

Element 206 (Access control):

This element, starting from the recognition result operated by element 204, performs the forwarding action associated to such recognition, or the refusal action associated to the failed recognition.

In the event of acceptance the communication frame will be forwarded to the server of reference.

In the event of refusal, the communication frame will be returned to the sender, together with possible explanations of the refusal. In fact, by virtue of the adopted <item>/<action> structure, it will be possible to associate actions, even complex ones as the construction of answer streams.

Element 207 (Access coordination):

This element, starting from the recognition result operated by element 204, performs the coordinating action associated to such recognition.

Such coordinating action relates to the individuation of the parameters to be forwarded to the server for the required coordination, individuation of the sender, formatting of the parameters to be forwarded, sending of parameters, acquisition of the answer from the server and forwarding of the obtained answer to element 204 for a possible prosecution of the recognition.

This approach is made possible by means of the second introduced notation, as by virtue of this notation actions can be associated, even complex ones as the construction of streams to be forwarded to specific network accessible servers. The coordinating element proves useful when the apparatus is used to manage communication among applications, therefore on high layer protocols (as those between client and server applications transferred on a TCP layer). In fact, in this event the apparatus, by virtue of the actions associated to input stream recognition, can operate changes in the stream for its re-forwarding to other application servers provided with different application protocols. A typical event occurs when the mutual operativity and the application cooperation have to be managed in a heterogeneous context, and where different "application servers" or different broker devices need to coexist (here referring also to the various CORBA -Common Object Request Broker Architecture- implementations, always not fully compatible among them) in presence of client applications often designed to converse using old application protocols.

The present invention has been described hereto with reference to one of its embodiments, given as non-limiting examples.

Furthermore, it is intended that there are other possible embodiments and kind of services falling within the protective scope of the present industrial property right.

CLAIMS

1. A network access control device through deterministic recognition of application frames satisfying a set of predetermined rules comprising:

- means (205) for monitoring and interpretation of the application frames to recognize;
- means (201) for storing predetermined rules;
- means (202) for compiling the predetermined rules in a direct access data structure;
- means (203)for storing said direct access data structure; and
- means (204) for comparing the application frames to be recognized with said direct access data structure,

wherein the recognition can be performed on any frame component and the direct access data structure allows an access time substantially independent from the number of rules.

2. The access control device according to claim 1, characterized in that said compiling means (202) of the predetermined rules comprise:

- conversion means, for converting the predetermined rules in a set of basic sequences of numerical identifiers; and
- compression means, for compressing the set of sequences thus obtained in a direct access data structure.

3. The access control device according to claim 1 or 2, characterized in that it further comprises forwarding means, for forwarding the application frame when recognized and return-to-sender means, for returning of the application frame when not recognized.

4. The access control device according to claim 3, characterized in that said return-to-sender means, for returning the application frames when not recognized, return information related to the reason of the failed forwarding..

5. The access control device according to any of the preceding claims, characterized in that the predetermined rules are stored as pairs of <object>/<action> fields.

6. The access control device according to claim 5, characterized in that the predetermined rules are stored as pairs of <data type>/<data value> fields.

7. The access control device according to claim 5 or 6, characterized in that the predetermined rules include one or more joker values.

8. The network access monitoring device according to claim 5, characterized in that the field <action> refers to the minimal set of commands

- Push
<value>
<variable>
<reading position>
<value at the reading position>
- Pop
<variable>
<reading position>
<at the reading position>
- And
- Mul
- Add
- Equal
- Next
- F_send_all
- F_dynamic.

9. The access control device according to claims 2 and 5, characterized in that the direct access data structure is represented through a matricial structure comprising object fields and action fields.

10. The access control device according to any of the preceding claims, characterized in that the means (205) for monitoring and interpretation of the application frames comprise:

a) a data packets monitoring device (9) at a layer corresponding to the OSI layer 2, said data packets comprising control frames and information frames, wherein the control and information frames contain a header portion and a body portion, said

header portion allowing the distinction between an information frame and a control frame;

- a control unit (15) receiving as an input the data coming from the monitoring device (9) and comprising means for the discrimination of the control frames from the information frames;
- a dating unit (16) connected to the control unit (15) and associating a monitoring time to the control frames and to the information frames;
- a discriminated data storing unit (17), storing the control and the information frames and the monitoring time thereof, bidirectionally connected to the control unit (15);
- a predetermined data storing unit (18), bidirectionally connected to the control unit (15), said predetermined data representing possible interpretations of the information frames contained in the discriminated data storing unit (17) and being apt to be compared, thorough the control unit (15), with the data contained in the body portion of the information frames or of the control frames, stored in the discriminated data storing unit (17), in such a way as to allow:
- an ordering, according to the time and kind of communication, of the body portions of the control frames and of the information frames;
- a reconstruction of application trees containing statistic information according to the kind of communication, thus certifying the communications and the monitoring of possible anomalies.

11. The access control device according to claim 10, characterized in that the data monitoring device (9) includes:

- a source data receiver (12);
- a destination data receiver (13);
- a connection interface (14) apt to receive the data coming from the source data receiver (12) and from the destination data receiver (13) and to transmit said data to the control unit (15).

12. The access control device according to claim 10 or 11, characterized in that the reconstruction of said application tree containing statistic information is obtained by reciprocal comparison of the body portion of the information frames.

13. The access control device according to any of the claims from 10 to 12, characterized in that the reconstruction of said application tree containing statistic

information is obtained by comparison of each sequence of body portions of the information frames with a set of predetermined sequences, said predetermined sequences representing possible interpretations of the information or control frames sequences contained in the discriminated data storing unit (17), said predetermined sequences being contained in said predetermined data storing unit (18).

14. The access control device according to any of the claims from 10 to 13, characterized in that said dating unit (16) is of the absolute time kind, in particular a radio or a satellite dating unit.

15. The access control device substantially as previously described with reference to the attached drawings.

On behalf of ALASI, di Arcieri Franco & C. s.a.s.

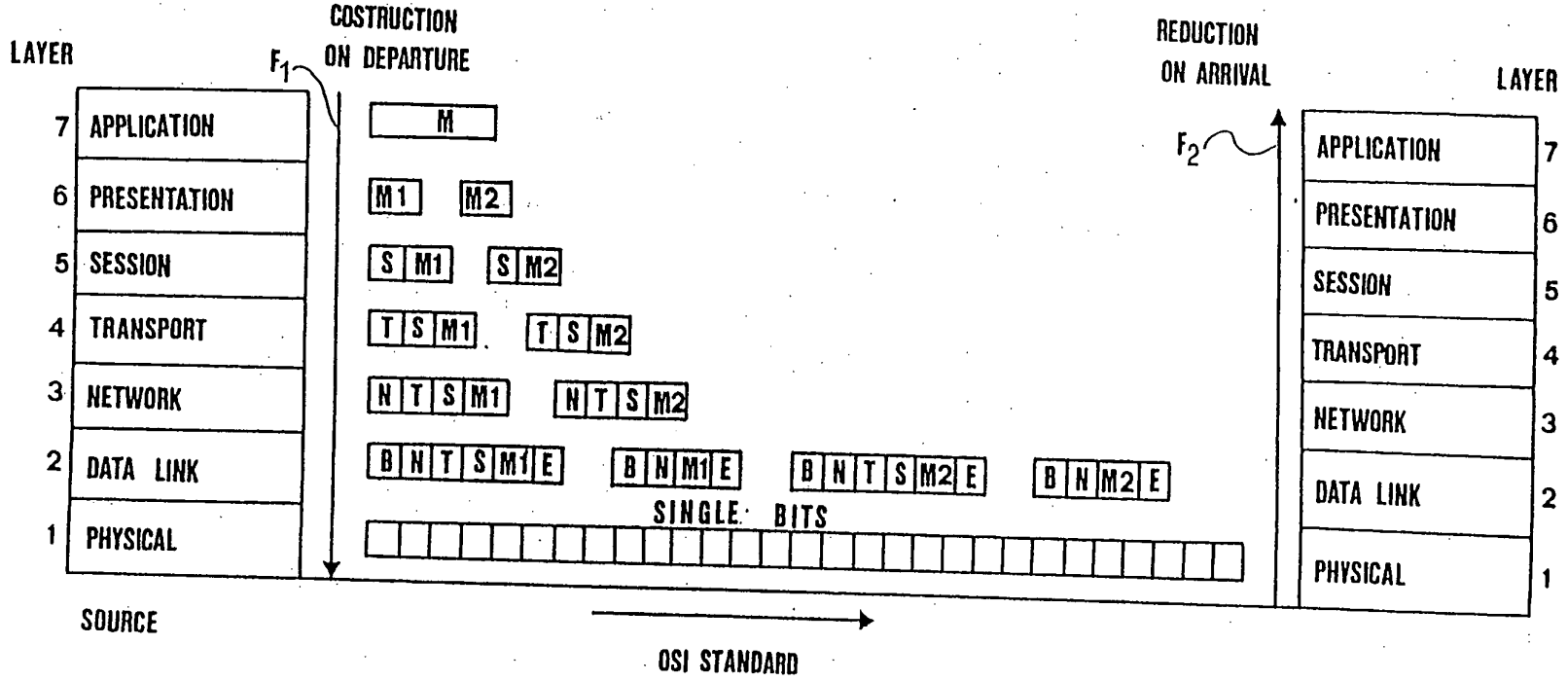

LAYER
COSTRUCTION
ON DEPARTURE
F1
REDUCTION
ON ARRIVAL
F2
LAYER
7 APPLICATION
6 PRESENTATION
5 SESSION
4 TRANSPORT
3 NETWORK
2 DATA LINK
1 PHYSICAL
M
M1
M2
S M1
S M2
T S M1
T S M2
N T S M1
N T S M2
B N T S M1 E
B N M1 E
B N T S M2 E
B N M2 E
SINGLE BITS
SOURCE
OSI STANDARD

FIG 1

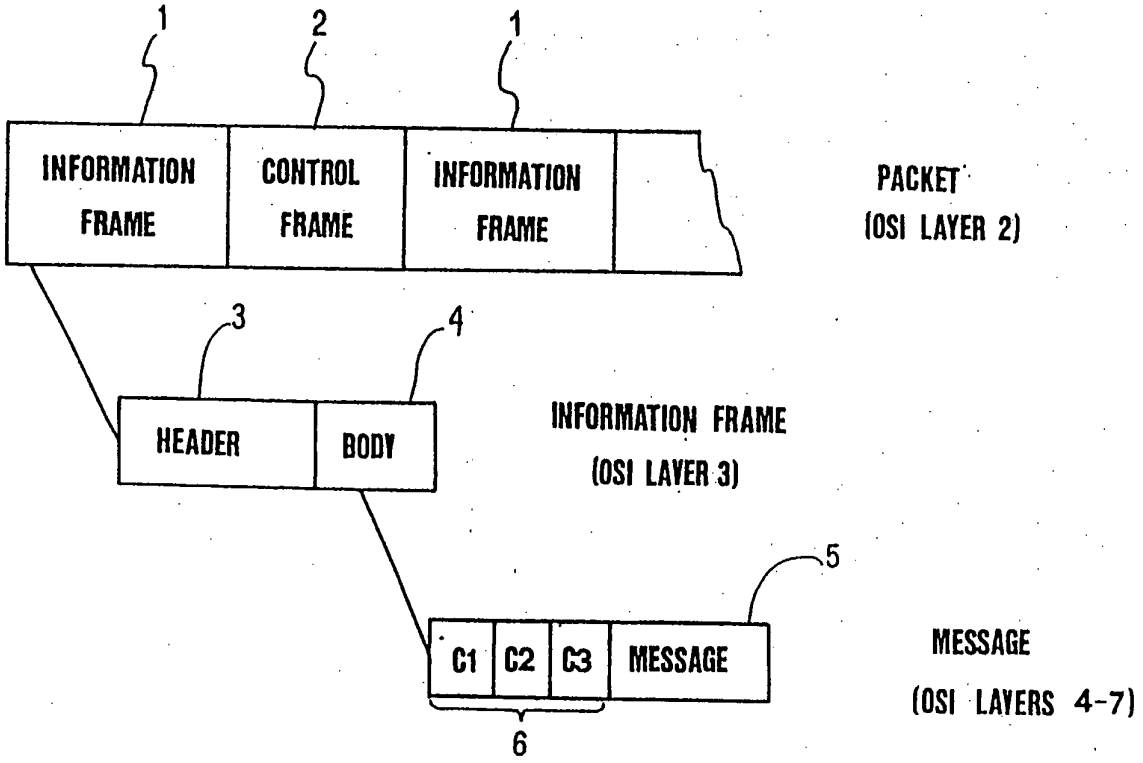

1
2
1
INFORMATION FRAME
CONTROL FRAME
INFORMATION FRAME
PACKET
(OSI LAYER 2)
3
4
HEADER
BODY
INFORMATION FRAME
(OSI LAYER 3)
5
C1
C2
C3
MESSAGE
MESSAGE
(OSI LAYERS 4-7)
6

FIG 2

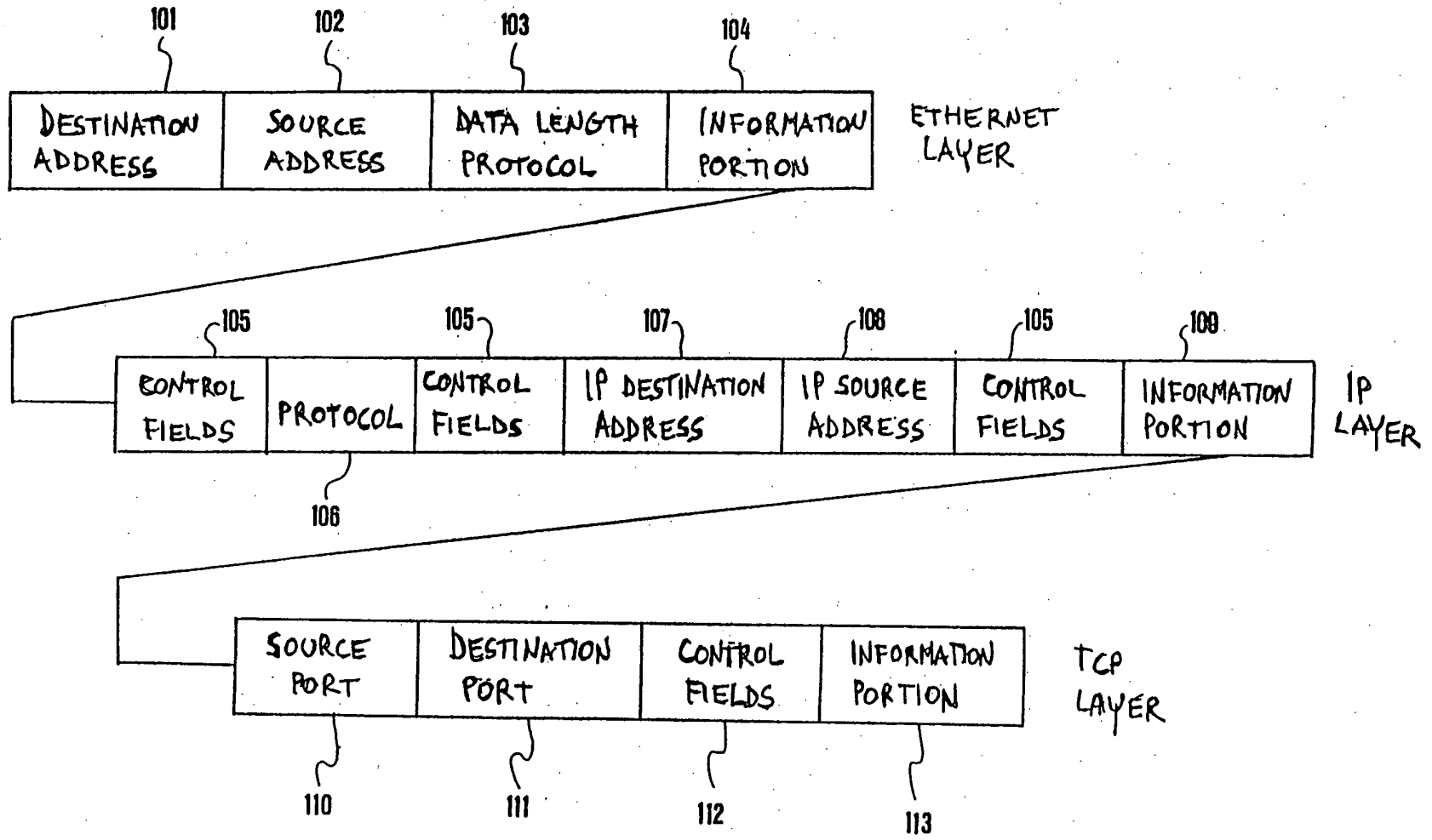

101
102
103
104
DESTINATION ADDRESS
SOURCE ADDRESS
DATA LENGTH PROTOCOL
INFORMATION PORTION
ETHERNET LAYER
105
105
107
108
105
109
CONTROL FIELDS
PROTOCOL
CONTROL FIELDS
IP DESTINATION ADDRESS
IP SOURCE ADDRESS
CONTROL FIELDS
INFORMATION PORTION
IP LAYER
106
SOURCE PORT
DESTINATION PORT
CONTROL FIELDS
INFORMATION PORTION
TCP LAYER
110
111
112
113

FIG.3

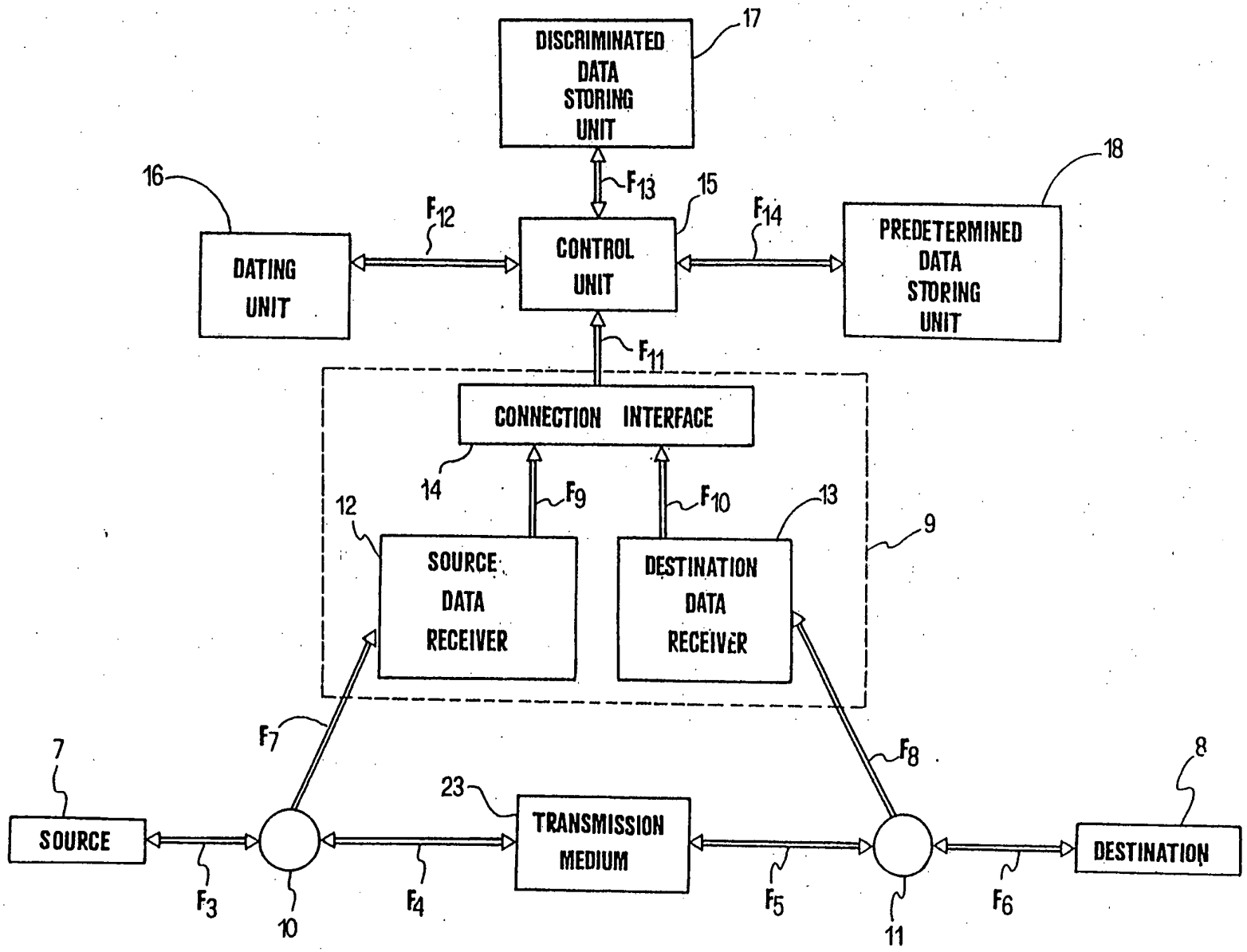

DISCRIMINATED DATA STORING UNIT
17
F13
15
18
16
F12
F14
DATING UNIT
CONTROL UNIT
PREDETERMINED DATA STORING UNIT
F11
CONNECTION INTERFACE
14
F9
F10
13
12
9
SOURCE DATA RECEIVER
DESTINATION DATA RECEIVER
F7
F8
7
8
23
SOURCE
TRANSMISSION MEDIUM
DESTINATION
F3
F4
F5
F6
10
11

FIG 4

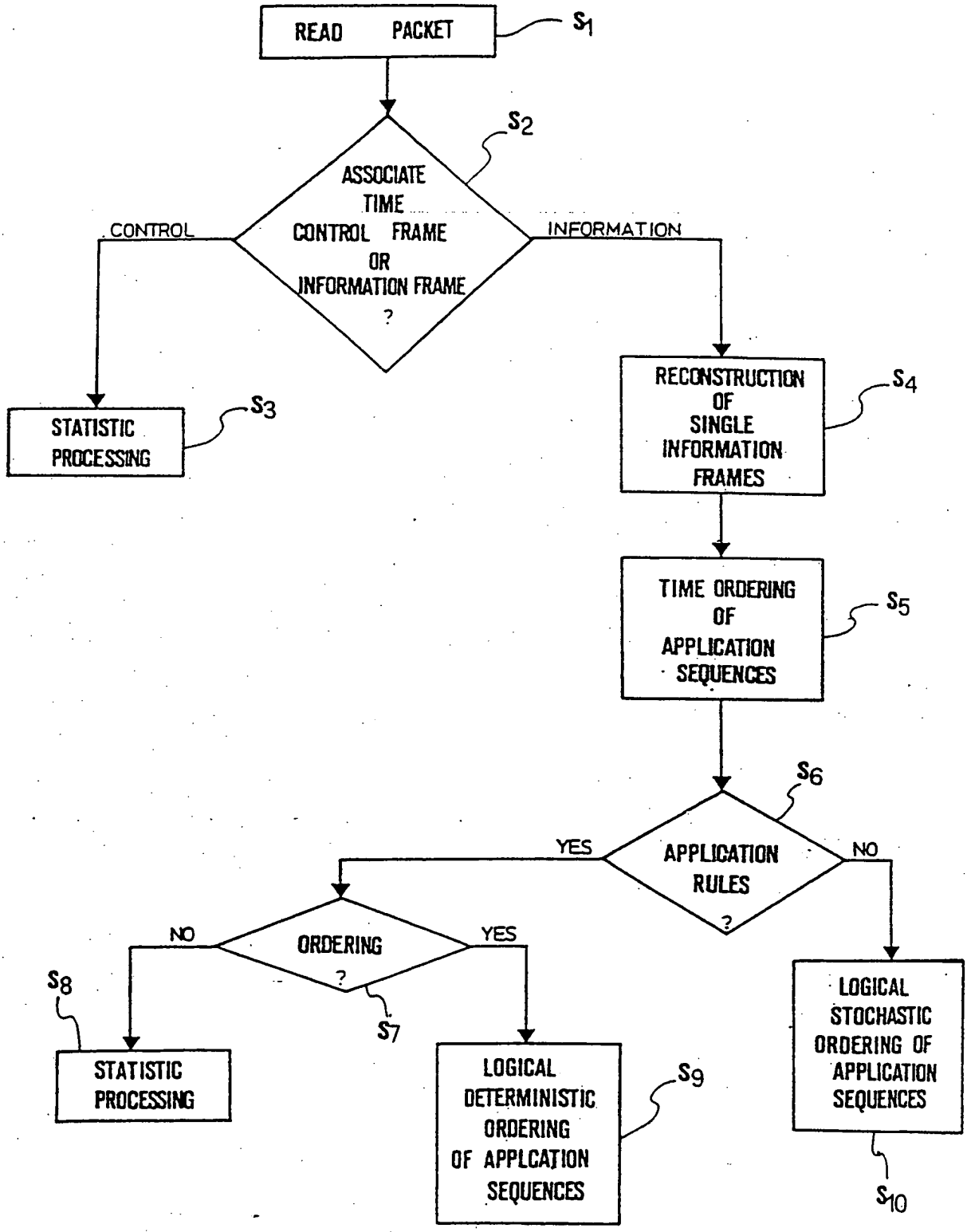

READ PACKET
S1
ASSOCIATE TIME CONTROL FRAME OR INFORMATION FRAME ?
S2
CONTROL
INFORMATION
STATISTIC PROCESSING
S3
RECONSTRUCTION OF SINGLE INFORMATION FRAMES
S4
TIME ORDERING OF APPLICATION SEQUENCES
S5
APPLICATION RULES ?
S6
YES
NO
ORDERING ?
S7
NO
YES
S8
STATISTIC PROCESSING
LOGICAL DETERMINISTIC ORDERING OF APPLCATION SEQUENCES
S9
LOGICAL STOCHASTIC ORDERING OF APPLICATION SEQUENCES
S10

FIG 5

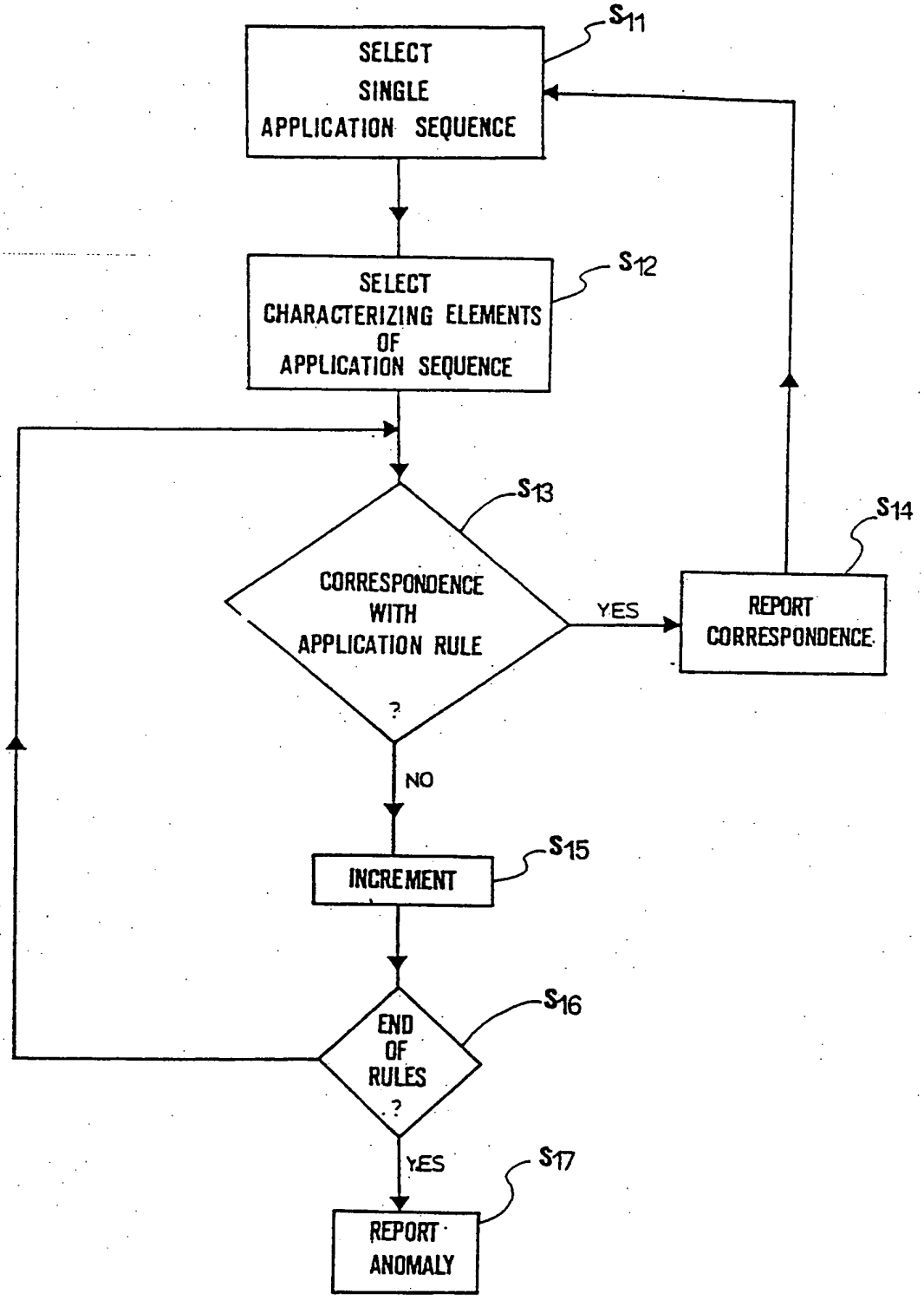

SELECT SINGLE APPLICATION SEQUENCE
S11
SELECT CHARACTERIZING ELEMENTS OF APPLICATION SEQUENCE
S12
CORRESPONDENCE WITH APPLICATION RULE ?
S13
YES
REPORT CORRESPONDENCE
S14
NO
INCREMENT
S15
END OF RULES ?
S16
YES
REPORT ANOMALY
S17

FIG 6

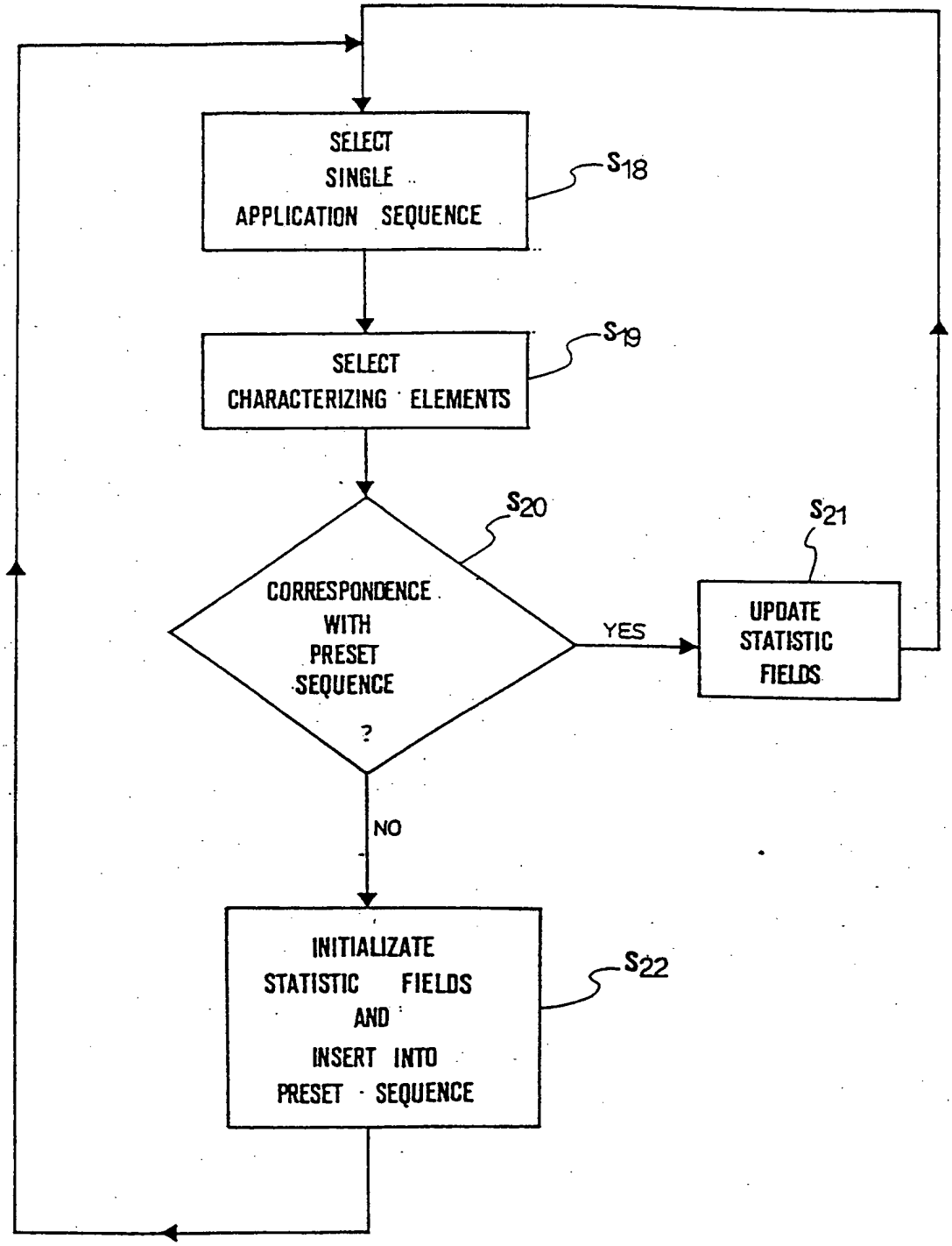

SELECT SINGLE APPLICATION SEQUENCE
S18
SELECT CHARACTERIZING ELEMENTS
S19
CORRESPONDENCE WITH PRESET SEQUENCE ?
S20
YES
UPDATE STATISTIC FIELDS
S21
NO
INITIALIZATE STATISTIC FIELDS AND INSERT INTO PRESET SEQUENCE
S22

FIG 7

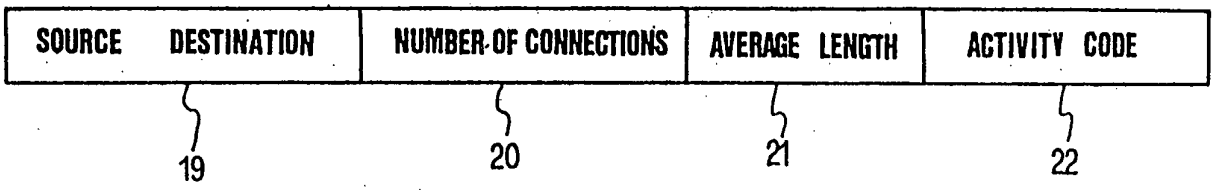

SOURCE DESTINATION
NUMBER OF CONNECTIONS
AVERAGE LENGTH
ACTIVITY CODE
19
20
21
22

FIG 8A

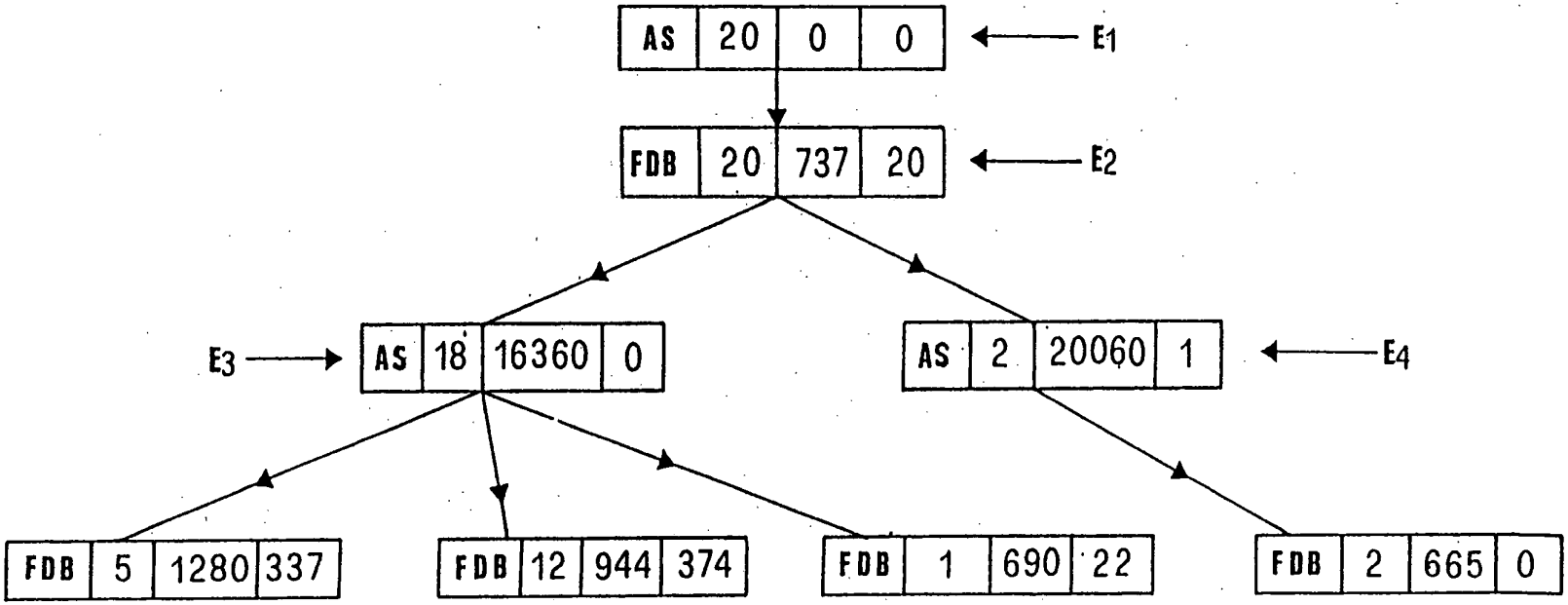

AS 20 0 0
E1
FDB 20 737 20
E2
E3
AS 18 16360 0
AS 2 20060 1
E4
FDB 5 1280 337
FDB 12 944 374
FDB 1 690 22
FDB 2 665 0

FIG 8B

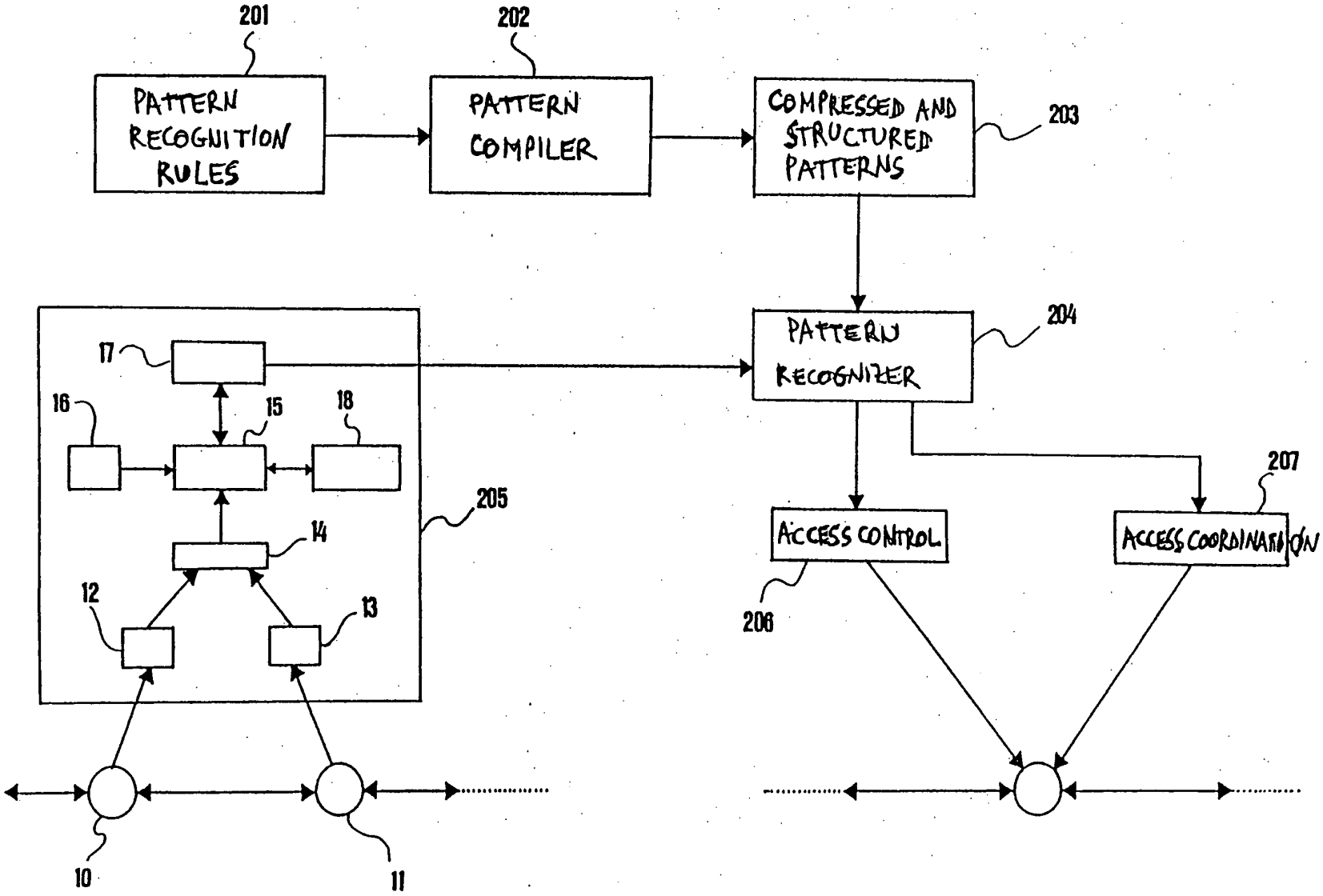

201
PATTERN RECOGNITION RULES
202
PATTERN COMPILER
COMPRESSED AND STRUCTURED PATTERNS
203
PATTERN RECOGNIZER
204
205
17
16
15
18
14
12
13
10
11
ACCESS CONTROL
206
207
ACCESS COORDINATION

FIG.9

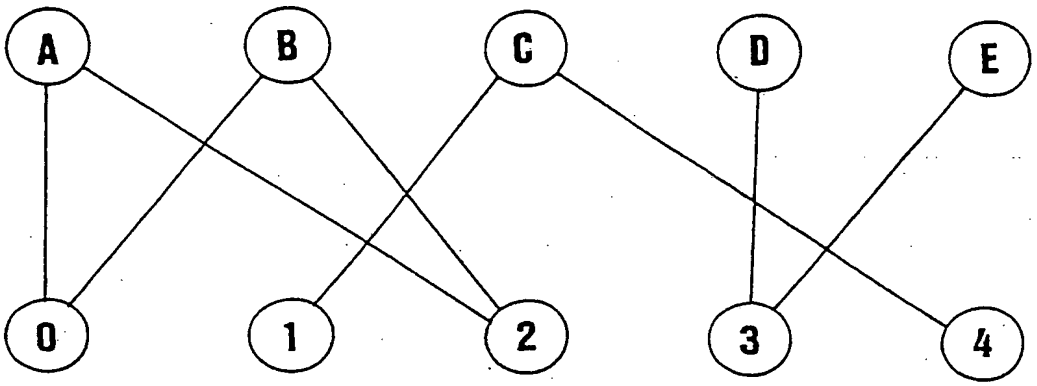


FIG.10A

	0	1	2	3	4
A	×		×		
B	×		×		
C		×			×
D			×		
E			×		

FIG. 10B

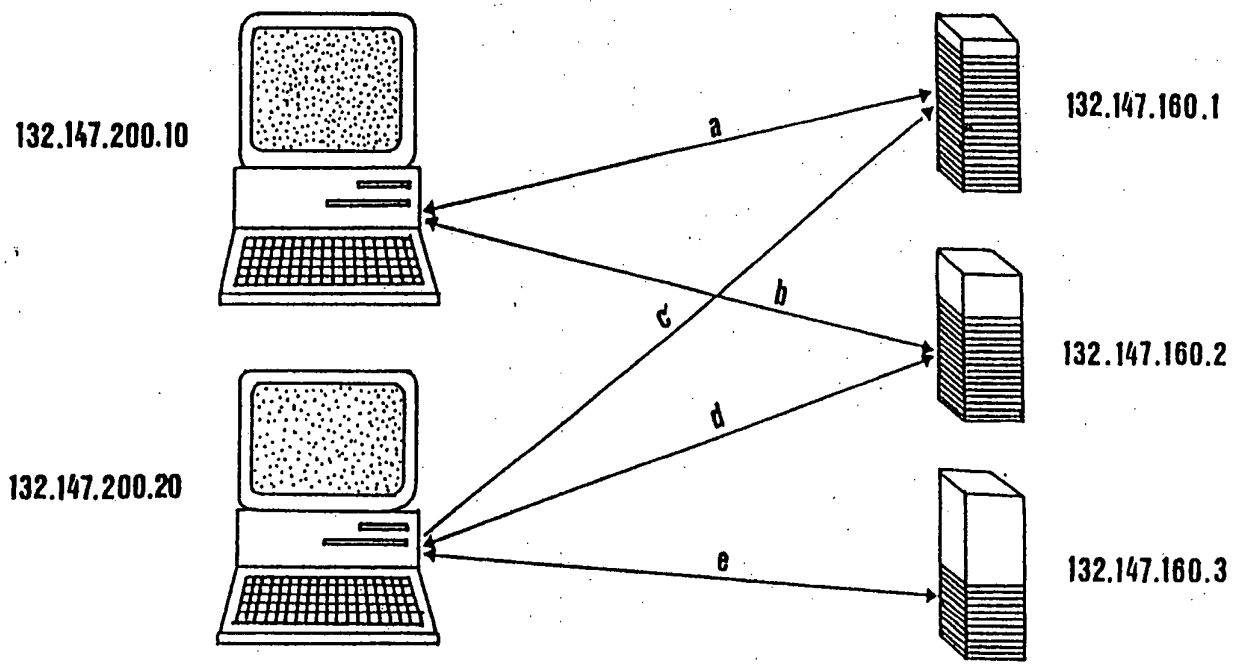

132.147.200.10
132.147.200.20
a
b
c
d
e
132.147.160.1
132.147.160.2
132.147.160.3

FIG.11

	0	1	2	3	4	5	6	7	8	9	0A	0B	0C
1:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(0A,400F)	(84,4010)	(93,4011)	(A0,4012)	(01,4013)	(00,8002)	(50,8003)
2:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(0A,400F)	(84,4010)	(93,4011)	(A0,4012)	(02,4013)	(00,8002)	(19,8003)
3:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(0A,400F)	(84,4010)	(93,4011)	(A0,4012)	(02,4013)	(00,8002)	(89,8003)
4:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(0A,400F)	(84,4010)	(93,4011)	(A0,4012)	(02,4013)	(00,8002)	(8A,8003)
5:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(0A,400F)	(84,4010)	(93,4011)	(A0,4012)	(02,4013)	(00,8002)	(8B,8003)
6:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(14,400F)	(84,4010)	(93,4011)	(A0,4012)	(01,4013)	(00,8002)	(14,8003)
7:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(14,400F)	(84,4010)	(93,4011)	(A0,4012)	(01,4013)	(00,8002)	(15,8003)
8:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(14,400F)	(84,4010)	(93,4011)	(A0,4012)	(01,4013)	(00,8002)	(17,8003)
9:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(14,400F)	(84,4010)	(93,4011)	(A0,4012)	(02,4013)	(00,8002)	(19,8003)
0A:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(14,400F)	(84,4010)	(93,4011)	(A0,4012)	(02,4013)	(00,8002)	(50,8003)
0B:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(14,400F)	(84,4010)	(93,4011)	(A0,4012)	(03,4013)	(00,8002)	(50,8003)
0C:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(14,400F)	(84,4010)	(93,4011)	(A0,4012)	(03,4013)	(00,8002)	(A1,8003)
0D:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(14,400F)	(84,4010)	(93,4011)	(A0,4012)	(03,4013)	(00,8002)	(A2,8003)
0E:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(14,400F)	(84,4010)	(93,4011)	(A0,4012)	(03,4013)	(08,8002)	(01,8003)
0F:	(08,000C)	(00,000D)	(06,4009)	(84,400C)	(93,400D)	(C8,400E)	(14,400F)	(84,4010)	(93,4011)	(A0,4012)	(03,4013)	(00,8002)	(17,8003)
10:	(08,000C)	(06,000D)											
11:	(08,000C)	(00,000D)	(01,4009)										

FIG.12